Ministerul Educației și Cercetării al Republicii Moldova  
Colegiul Universității Tehnice al Moldovei

**RAPORTUL  
STAGIULUI DE PRACTICĂ**

**TEMA:** Aplicație de comparare prețuri între magazine

Elevul: **Andriuță Vlad**

Grupa: **PTPP-241**Specialitatea: **Programarea si Testarea Produselor de Program**

Baza de practică: **CUTM, Chisinau**

Conducătorul stagiului de practică de la  
Colegiul Universității Tehnice a Moldovei  
**Moraru Magdalena**

Chișinău 2025

**Ziua 1**

**Rezumat**

Scopul acestei aplicații este de a sprijini procesul de cumpărare, prin compararea rapidă a prețurilor pentru aceleași produse disponibile în mai multe magazine. Utilizatorul beneficiază de o interfață simplă și intuitivă, care afișează automat prețurile actuale pentru produse esențiale, precum lapte, pâine, ulei, oua, carne sau orice alte produse, în diferite supermarketuri. Astfel, poate observa dintr-o privire care magazin oferă cel mai bun preț si care este cel mai scump. Datele sunt prezentate într-un tabel comparativ, iar aplicația evidențiază vizual cel mai ieftin preț și cel mai scump ,pentru a ușura procesul decizional. Prețurile sunt simulate, fiind introduse manual în cod sau încărcate dintr-un fișier extern, imitând o sursă reală de informații. Această aplicație a fost concepută cu scop educațional, utilizând limbaje web precum HTML, CSS și JavaScript. Ea dezvoltă gândirea logică, abilitățile de lucru cu date structurate și capacitatea de a crea interfețe prietenoase. Proiectul simulează un scenariu des întâlnit în viața reală și se inspiră din aplicații cunoscute precum Compari.ro sau Google Shopping, având potențialul de a fi extins cu funcționalități precum filtre, căutare sau afișarea de imagini.

**Introducere**

Ai un produs în minte? Scrie-l mai jos și noi îți arătăm unde e **mai ieftin**, **mai rapid** și **mai simplu**.  
Nu mai pierde timp să cauți prin zeci de magazine – le-am pus pe toate într-un singur loc.

Cauți. Compari. Cumperi inteligent.

1. **ANALIZA DOMENIULUI DE STUDIU**

**1.2 Sisteme similare cu proiectul realizat:**

**1.Compari.ro**

**2.Price.ro**

**3.Idealo.de**

**4.Pricerunner.com**

**5.Pricespy.co.uk**

**Avantaje:**

**1. Compari.ro**

* Gama largă de produse și magazine.
* Filtre detaliate pentru căutare.
* Usor de interactionat

**2. Compari.ro**

* Actualizarea frecventa a datelor
* Multe criterii dupa care poate fi filtrate
* Oferte promotionale vizibile

**3. Idealo.de:**

* Design modernsi placut
* Actualizarea frecventa a datelor
* **Comparații clare între prețuri**:

**Ziua 2:**

În cadrul lecției, s-a învățat cum să folosească Git și GitHub pentru a salva și organiza proiectele de programare. Git este un sistem de control al versiunilor care ajută la urmărirea modificărilor dintr-un proiect, iar GitHub este o platformă online unde proiectele pot fi încărcate pentru a fi păstrate în siguranță sau pentru a colabora cu alți colegi.

Mai întâi, a fost creat un cont pe GitHub, completându-se emailul, parola și confirmându-se un cod trimis pe email. După autentificare, a fost creat un repozitoriu nou, adică o mapă digitală în care vor fi salvate fișierele proiectului.

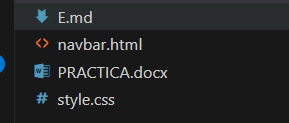
Apoi, a fost instalat Git pe calculator. Instalarea a fost verificată cu comanda git --version, iar apoi s-au setat numele și emailul pentru ca Git să știe cine face modificările. În folderul proiectului, a fost rulată comanda git init pentru a crea un repozitoriu local. După aceea, fișierele au fost adăugate folosind comanda git add ., modificările au fost salvate cu comanda git commit -m "mesaj", iar proiectul local a fost conectat cu cel de pe GitHub prin link-ul repozitoriului. În final, fișierele au fost urcate online cu comanda git push.

Git ajută la prevenirea pierderii progresului, permite revenirea la versiuni anterioare și face colaborarea în echipă mai ușoară. Este utilizat frecvent în companiile IT, dar și în școli sau proiecte personale.

Git se folosește împreună cu un editor de cod unde pot fi scrise programele de exemplu Visual Studio Code (VSCode). VSCode are integrare cu Git, astfel că se pot face commit și push direct din interfața grafică, fără a mai fi nevoie de comenzi.

În concluzie, Git și GitHub sunt instrumente esențiale în programare. Ele ajută la organizarea proiectelor, la creșterea eficienței și la salvarea muncii într-un mod profesionist. Chiar dacă pare complicat la început, totul devine mai clar după puțină practică.

****

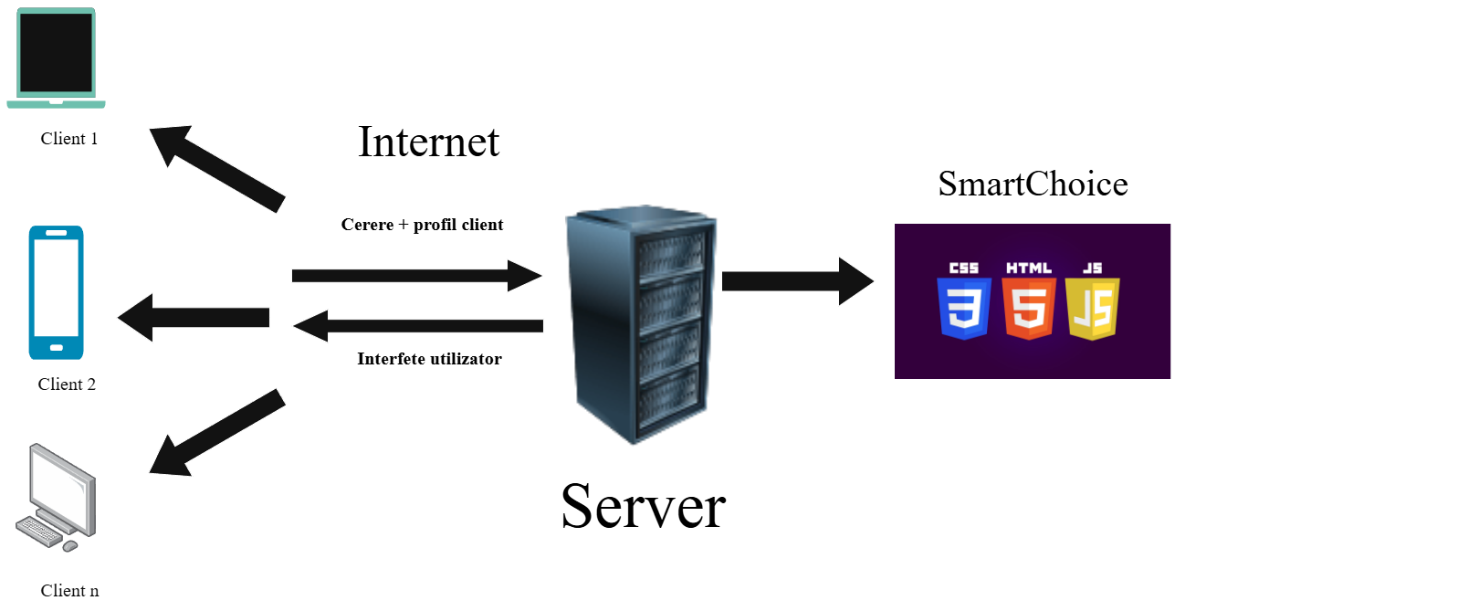
** **

**Ziua 3:**

***a. Interacțiunea dintre client și server***

* Ce este un client, ce este un server, cum interactionează(realizarea unei scheme utilizand  tool-ul: [https://app.diagrams.net/?src=about](https://app.diagrams.net/?src=about" \t "_blank) despre modul de interactiune dintre client si server);
* Un **client** este o aplicație care solicită un serviciu de la o altă aplicație, de obicei aflată pe un server
* Un **server** este o aplicație sau calculator care oferă servicii utilizatorilor internetului. Este un program care primește cereri, îndeplinește serviciul cerut și transmite rezultatul drept răspuns. Un server poate deservi mai mulți clienți în același timp.
* Interacțiunea dintre client și server:

Clientul trimite o cerere către server, cerere care este procesată, iar răspunsul este trimis înapoi. Această comunicare se face printr-un protocol de rețea, cum ar fi TCP/IP.



* **Ce este HTTP și HTTPS, pentru ce avem nevoie, ce este un protocol(analiza protocoalelor TCP si UDP). Ce este un port? Ce port au protocoalele HTTP, HTTPS, TCP, UDP.**
* **Protocolul** este un set de reguli care permite calculatoarelor să comunice între ele.
* **HTTP (HyperText Transfer Protocol)** este un protocol folosit pentru accesarea paginilor web, dar **nu este securizat** – datele pot fi interceptate.
* **HTTPS (HyperText Transfer Protocol Secure)** este versiunea securizată a HTTP. Folosește criptare, protejând informațiile transmise.
* **TCP (Transmission Control Protocol)** este un protocol care asigură transmiterea corectă și ordonată a datelor între două dispozitive. Verifică dacă datele ajung complet și în ordine, iar dacă ceva se pierde, le retrimite. Este mai sigur, dar mai lent. Se folosește la: pagini web (HTTP/HTTPS), e-mail, fișiere descărcate etc.
* **UDP (User Datagram Protocol)** este un protocol care trimite datele rapid, fără să verifice dacă ajung corect. Este mai puțin sigur, dar mult mai rapid, fiind folosit în aplicații unde viteza este mai importantă decât acuratețea, cum ar fi jocurile online sau apelurile video. Se folosește la: apeluri video/audio, jocuri online.
* **Portul** este un identificator numeric folosit în rețelele de calculatoare pentru a diferenția mai multe servicii sau aplicații care rulează pe un singur dispozitiv. Porturile funcționează împreună cu adresa IP pentru a direcționa traficul de rețea către aplicația corectă.
* **HTTP** – port **80**

**HTTPS** – port **443**

**TCP** – folosește porturi diverse (ex: 80, 443, 22, etc.)

**UDP** – folosește porturi diverse (ex: 53, 67, 68, etc.)

* **b. Arhitectura aplicației web-based:**
* **Aplicația web-based** este un program software care rulează pe un server și este accesat printr-un browser web, folosind internetul sau o rețea locală.
* Componentele principale:

 **Client (Front-end)**

* Partea vizibilă pentru utilizator, interfața grafică.
* Tehnologii: **HTML, CSS, JavaScript, framework-uri**

 **Server (Back-end)**

* Procesează cererile de la client, gestionează logica aplicației.
* Tehnologii: **Node.js, PHP, Python, Java, Ruby**, etc.

 **Bază de date**

* Păstrează datele aplicației
* Exemple: **MySQL, PostgreSQL, MongoDB**

 **Server web**

* Intermediază comunicarea între client și aplicația de pe server.
* Exemple: **Apache, Nginx**

 **Protocol de comunicare**

1. Folosit pentru schimbul de date între client și server.
2. Exemple: **HTTP, HTTPS**

* **Avantajele unei  arhitecturi web:**

  Accesibilitate de oriunde

 Fără instalare locală

 Actualizări centralizate

 Costuri mai mici de întreținere

 Compatibilitate multiplatformă

 Securitate centralizată

|  |  |
| --- | --- |
| **Frontend** | **Backend** |
| Reprezintă interfața vizibilă pentru utilizator. | Reprezintă **partea invizibilă**, care procesează datele și logica aplicației |
| Se ocupă cu **designul, interacțiunea și afișarea conținutului** | Se ocupă cu **autentificare, baze de date, API-uri, securitate** |
| Tehnologii folosite: **HTML, CSS, JavaScript,framework-uri** | Tehnologii folosite: **Node.js, Python, PHP, Java,** baze de date |

|  |  |  |
| --- | --- | --- |
| **Caracteristică** | **Arhitectură Monolitică** | **Arhitectură pe Microservicii** |
| Structură | O singură aplicație unitară | Aplicația este împărțită în servicii mici și independente |
| Dezvoltare | Dezvoltare într-un singur cod sursă | Fiecare microserviciu poate fi dezvoltat separat |
| Scalabilitate | Scalare dificilă, doar pe întreaga aplicație | Scalare ușoară, doar pe serviciul necesar |
| Deploy | |  | | --- | | Deploy complet al aplicației |  |  | | --- | |  | | Deploy separat pentru fiecare serviciu |
| Timp de lansare | Mai lung, din cauza interdependențelor | Mai rapid, fiecare echipă lansează independent |
| |  | | --- | | Toleranță la erori |  |  | | --- | |  | | O eroare poate afecta toată aplicația | O eroare afectează doar serviciul respectiv |
| |  | | --- | | Tehnologii |  |  | | --- | |  | | De obicei, o singură tehnologie | Fiecare serviciu poate avea tehnologia proprie |
| Complexitate | Cod și structură mai simple | Mai complex de gestionat și orchestrat |



* **Directive uzuale** **HTML:**

<html> - Marchează începutul și sfârșitul unui document HTML.

<head> - Conține informații despre pagină, cum ar fi titlul, meta date, linkuri către CSS sau

scripturi.

<body> - Conține tot conținutul vizibil al paginii

<title> - Specifică titlul paginii care apare în bara de titlu a browserului sau în tab.

<p> - Creează un **paragraf** de text.

<h1>–<h6> - Creează titluri ,de la cel mai mare (<h1>) la cel mai mic (<h6>)

<br> - Inserează o **linie nouă** în text.

<hr> - Inserează o **linie orizontală** de separare.

Sunt folosite pentru structurarea conținutului unei pagini web

* **Ancore:**

<a> - Se folosesc pentru a crea **linkuri**.

* **Imagini:**

<img> - Folosite pentru a afișa imagini în pagină.

* **Culori de fundal:**

< body style = “background-color:(culoare);”>

* **Meta tag-uri:**

<meta…> - Folosite în secțiunea <head> pentru a furniza informații despre pagină



* **Formularele** HTML sunt folosite pentru a colecta date de la utilizatori:

<input> – pentru câmpuri de text, parole, butoane radio, checkbox-uri etc

<select> – pentru liste derulante.

<button> – pentru a trimite formularul.

* **Tabelele** sunt folosite pentru a afișa informațiile într-un format structurat, pe rânduri și coloane:

<table> – definește tabelul.

<tr> – definește un rând.

<td> – definește o celulă de date.

<th> – definește un antet de coloană.

* **Cadrele (frames)** erau folosite în versiunile mai vechi de HTML pentru a împărți fereastra browserului în mai multe secțiuni independente, fiecare cu propriul conținut. Frames nu mai sunt recomandate în HTML5 – sunt considerate învechite. În locul lor, se folosesc iframe-uri.