Realizat de : Berbece Razvan – Antonio

Clasa : XII B

Coordonator : Costrachevici Elena

Colegiul National de Informatica Piatra Neamt, Mai 2019

Cuprins

1. Motivatia temei alese

Pentru a putea fi acceptat in lumea profesionala a tehnologiei, un individ trebuie sa isi demonstreze capacitatile de a intelege conceptele IT-ului si de aplicare a acestora. Initial, intentionam sa realizez un CV intr-o forma neobisnuita, doar ca apoi sa realizez ca asta este o oportunitate de a realiza un portofoliu digital.

Acesta presupune o prezentare succinta a mea, a proiectelor la care am lucrat de-a lungul timpului, a problemelor intampinate in realizarea acestora si a solutiilor gasite, a abilitatilor si a tehnologiilor pe care le folosesc si un CV, alaturi de datele mele de contact.

Design-ul este cat mai simplu, observand ca “la moda” este minimalismul: folosirea albului sau a negrului, structura simpla, lipsa animatiilor evidente etc. .

2. Informatii de baza ale tehnologiilor folosite

HTML nu este un limbaj de programare; este un *limbaj de marcare* care definește structura conținutului tău. HTML este alcătuit dintr-o serie de **elements**, pe care le închizi sau în care încadrezi diferite părți ale conținutului pentru a face să apară într-un anumit mod sau pentru a se comporta într-un anumit fel. tags de închidere pot face dintr-un un cuvânt sau o imagine un link către un alt loc, pot italiza cuvinte, și pot face ca dimensiunea fontului să fie mai mare sau mai mică, etc. De exemplu, ia în considerare următoarea linie de conținut:

```
My cat is very grumpy
```

Dacă vrei ca linia să iasă în evidență, ai putea specifica că este un paragraf prin includerea acestuia în taguri de paragraf:

```
<p>My cat is very grumpy</p>
```

Principalele părți ale elementului nostru sunt:

1. **Tagul de deschidere:** Acesta constă din numele elementului (în acest caz, p), înconjurat de **paranteze unghiulare** de deschidere și închidere. Acestea indică locul în care elementul începe, sau începe să aibă efect — în acest caz unde începe paragraful.
2. **Tagul de închide:** Acesta este aceeași cu tagul de deschidere, cu excepția faptului că include un *slash* înainte de numele elementului. Acesta indică locul în care elementul se termină — în acest caz unde se

termină paragraful. Absența includerii unui tag de închidere este una dintre erorile comune de începător și poate duce la rezultate ciudate.
3. **Conținutul:** Acesta este conținutul elementului, care în acest caz este doar text.
4. **Elementul:** Tagul de deschidere, tagul de închidere și conținutul împreună cuprind elementul.

Anatomia unui document HTML

Acest lucru încheie elementele de bază HTML, dar acestea nu sunt utile singure. În continuare vei analiza modul în care elementele individuale sunt combinate pentru a forma o întreagă pagină HTML. Să revedem codul pe care l-am pus în exemplul index.html (pe care l-ai întâlnit pentru prima dată în articolul Gestionarea fișierelor):

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <img src="images/firefox-icon.png" alt="My test image">
  </body>
</html>
```

Aici avem:

- <!DOCTYPE html> — doctype. În negura timpului, când HTML era la început (aproximativ 1991/2), elementele doctype au fost menite să acționeze ca linkuri către un set de reguli pe care pagina HTML trebuia să le urmeze pentru a fi considerat codul HTML bun ceea ce ar putea înseamna verificarea automată a erorilor și alte lucruri utile. În orice caz, în zilele noastre nimănui nu-i mai pasă de ele, acestea fiind doar un artefact istoric care trebuie să fie inclus pentru ca totul să funcționeze corect. Pentru moment, asta este tot ce trebuie să știi.
- <html></html> — elementul <html>. Acest element include tot conținutul pe întreaga pagină, și este uneori cunoscut ca elementul root.
- <head></head> — elementul <head>. Acest element se comportă ca un container pentru toate lucrurile pe care dorești să le incluzi în pagina HTML, care *nu este* conținutul pe care îl afișezi vizitatorilor paginii tale. Acesta include lucruri precum keywords și o descriere a paginii care vrei să apară în rezultatele de căutare, CSS pentru a stiliza conținutul tău, declarațiile setului de caracter, și altele.
- <body></body> — elementul <body>. Acesta conține *tot* conținutul care vrei să-l afișezi utilizatorilor web atunci când vizitează pagina ta, indiferent dacă acesta este text, imagini, videoclipuri, jocuri, înregistrări audio sau orice altceva.
- <meta charset="utf-8"> — acest element stabilește setul de caractere pe care documentul tău ar trebui să-l folosească UTF-8, care include majoritatea caracterelor din marea majoritate a limbilor scrise de om. În esență, acesta poate manipula orice conținut text pe care l-ai putea introduce. Nu există niciun motiv pentru a nu seta acest lucru, și poate ajuta la evitarea unor probleme mai târziu.
- <title></title> — elementul <title>. Acesta setează titlul paginii tale, care este titlul care apare în fila browserului în care este încărcată pagina. Este, de asemenea, folosit pentru a descrie pagina atunci când o marchezi cu stea Bookmark, sau o adaugi la favorite.

Ce este CSS?

La fel ca HTML, CSS nu este chiar un limbaj de programare. Nu este nici *limbaj de marcare* — este un limbaj de *stilizare*. Asta înseamnă că îți permite să aplici selectiv stiluri elementelor din documentele HTML. De exemplu, pentru a selecta **toate** elementele paragraf dintr-o pagină HTML și a transforma textul din interiorul acestora în roșu, vei scrie acest cod CSS:

```
p {
  color: red;
}
```

Să încercăm: inserează aceste trei linii de CSS într-un fișier nou în editorul tău de text, și apoi salvează fișierul ca style.css în directorul tăustyles.
Dar mai trebuie să aplici codul CSS în documentul tău HTML. În caz contrar, stilul CSS nu va modifica modul în care browserul tău afișează documentul HTML (Dacă nu ai urmărit până acum proiectul nostru, citește Gestionarea fișierelor și Noțiuni de bază HTML pentru a afla ce trebuie să faci mai întâi).

1. Deschide fișierul index.html și inserează următoarea linie undeva în elementul head (adică între tagurile <head> și </head>):

```
<link href="styles/style.css" rel="stylesheet" type="text/css">
```

2. Salvează fișierulindex.html și încarcă-l în browserul tău. Ar trebui să vezi ceva asemănător:

My test page

file:///Users/chrismills/Dropbc

Google

Most Visited | MDN | Mozilla general | FirefoxOS ecosy… | Meeting details

Mozilla is cool

At Mozilla, we're a global community of

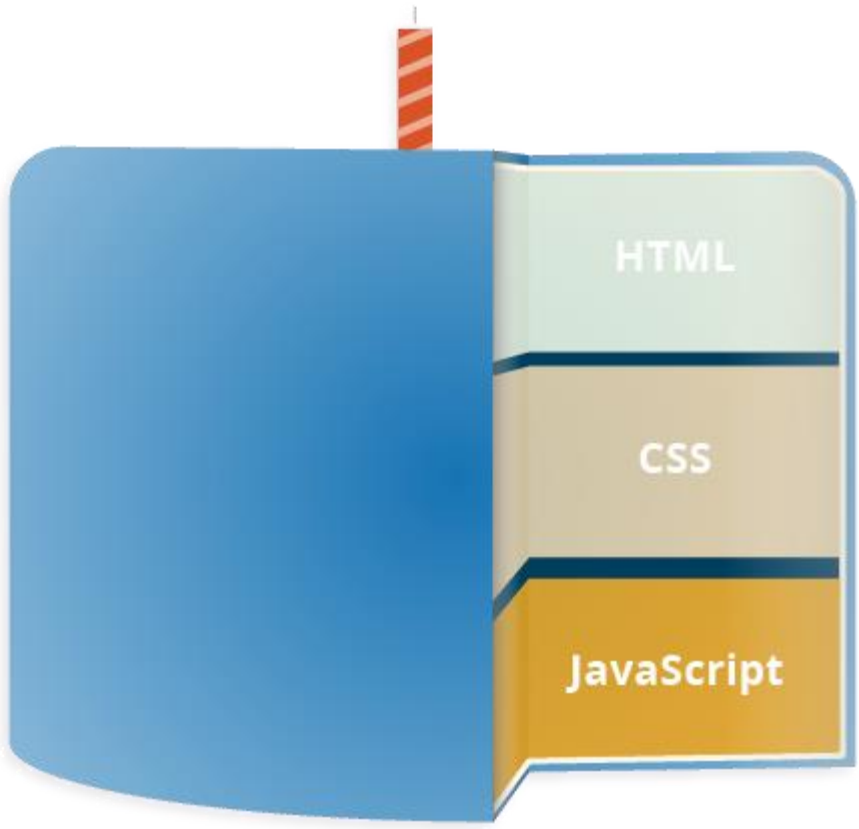
- technologists
- thinkers
- builders

working together to keep the Internet alive and accessible, so people worldwide can be informed contributors and creators of the Web. We believe this act of human collaboration across an open platform is essential to individual growth and our collective future.

Read the Mozilla Manifesto to learn even more about the values and principles that guide the pursuit of our mission.

O definiție de nivel înalt

JavaScript este un limbaj de programare care îți permite să implementezi lucruri complexe pe paginile web — de fiecare dată când o pagină web face mai mult decât să afișeze informații statice pentru ca tu să te uiți — afișând actualizări de conținut în timp util sau hărți interactive, grafice animate 2D/3D, video jukebox, etc. — poți să pariezi probabil că JavaScript este implicat. Este cel de-al treilea layer al tortului de layere de tehnologii web standarde, dintre care două (HTML și CSS) le-am acoperit în detaliu în alte părți ale Zonei de Învățare.



- HTML este limbajul de marcare pe care îl folosim pentru a structura și a da sens conținutului nostru web, de exemplu, definind paragrafe, titluri și tabele de date, sau încadrând în pagină imagini și videoclipuri.

- CSS este limbajul regulilor de stil pe care le utilizăm pentru a aplica stilul conținutul nostru HTML, de exemplu setarea culorilor de fundal și a fonturilor, și aranjarea conținutului nostru în mai multe coloane.
- JavaScript este un limbaj de scripting care îți permite să creezi conținut dinamic actualizat, de a controla multimedia, de a anima imagini și aproape orice altceva (Bine, nu totul, dar este uimitor ceea ce poți obține cu câteva linii de JavaScript).

  Cele trei layere se construiesc unul peste altul frumos. Să luăm ca exemplu o simplă etichetă de text. Putem marca acest lucru folosind HTML pentru a da structură și scop:

```
<p>Jucătorul 1: Chris</p>
```

Player 1: Chris

Apoi putem adăuga câteva linii de CSS în completare pentru a-l face să arate bine:

```
p {
  font-family: 'helvetica neue', helvetica, sans-serif;
  letter-spacing: 1px;
  text-transform: uppercase;
  text-align: center;
  border: 2px solid rgba(0,0,200,0.6);
  background: rgba(0,0,200,0.3);
  color: rgba(0,0,200,0.6);
  box-shadow: 1px 1px 2px rgba(0,0,200,0.4);
  border-radius: 10px;
  padding: 3px 10px;
```

```
  display: inline-block;
  cursor:pointer;
}
```

Și, în sfârșit, putem adăuga JavaScript pentru a implementa comportamentul dinamic:

```
var para = document.querySelector('p');

para.addEventListener('click', updateName);

function updateName() {
  var name = prompt('Enter a new name');
  para.textContent = 'Player 1: ' + name;
}
```

Încearcă să dai clic pentru a vedea ce se întâmplă (reține de asemenea, că poți găsi acest demo pe GitHub — vezi codul sursă sau rulați-l live)! JavaScript poate face mai mult decât atât — să explorăm mai în detaliu.

Ce poate face cu adevărat?

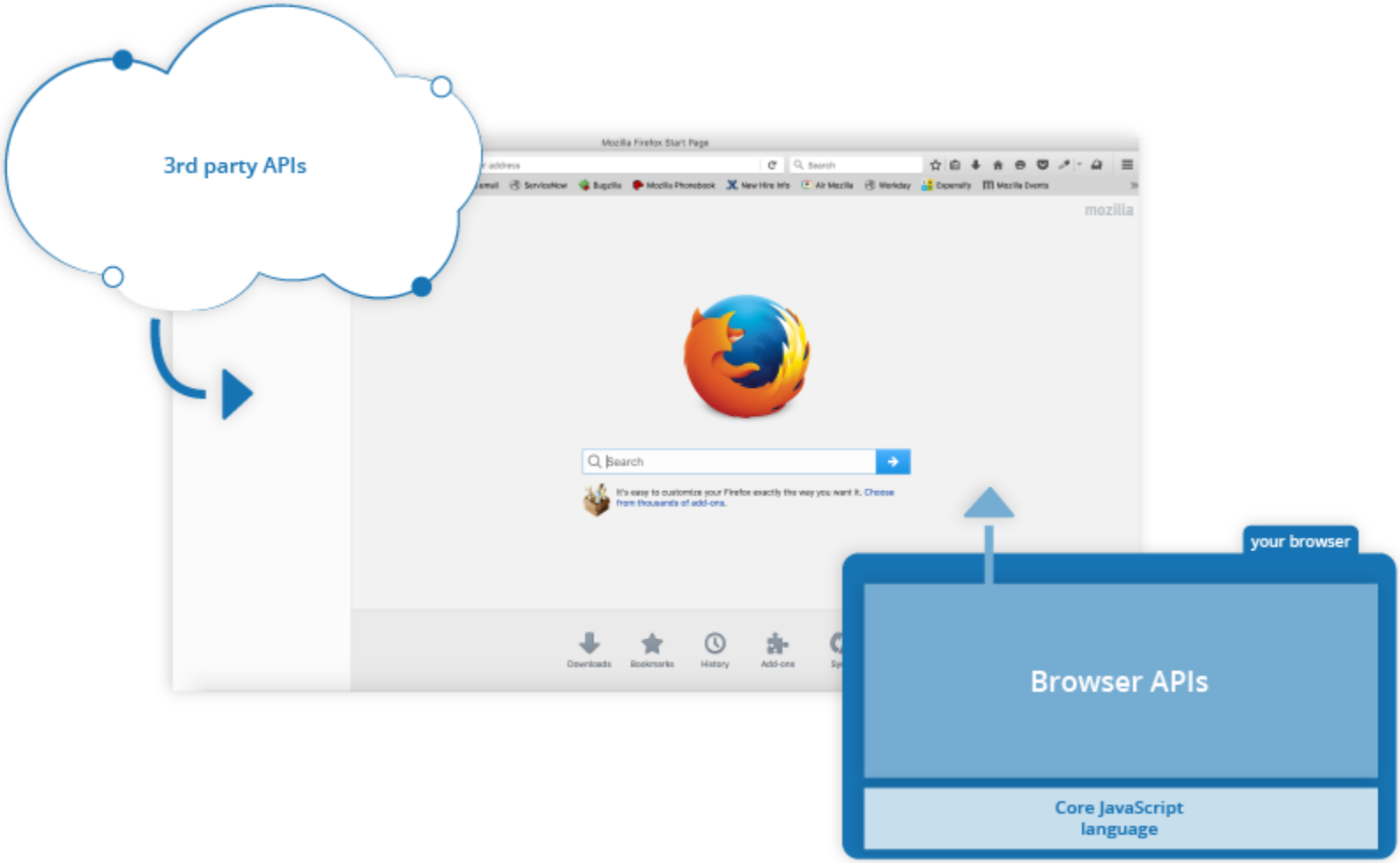
Limbajul JavaScript de bază constă în câteva funcții comune care îți permit să faci lucruri precum:

- Stochezi valori utile în interiorul variabilelor. În exemplul de mai sus de exemplu, solicităm introducerea unui nume nou și după îl salvăm într-o variabilă numită name.
- Operații pe fragmente de text (cunoscute ca „șiruri de caractere” în programare). În exemplul de mai sus, am luat șirul „Jucător 1: ” și l-am

adăugat la variabila name pentru a crea eticheta completă de text, de exemplu „Jucător 1: Chris”.

- Rularea codului ca răspuns la anumite evenimente care apar pe o pagină web. Am folosit evenimentul click din exemplul de mai sus pentru a detecta momentul în care butonul este apăsat și după se execută codul care actualizează eticheta de text.
- Și mult mai mult!

  Ceea ce este și mai interesant este totuși funcționalitatea construită deasupra limbajului de bază JavaScript. Așa numitele **Interfețe de programare a aplicațiilor (API)** oferă superputeri suplimentare pentru a le utiliza în codul tău JavaScript.
  APIurile sunt seturi de blocuri de coduri gata făcute care permit unui dezvoltator să implementeze programe care altfel ar fi greu sau imposibil de implementat. Acestea fac același lucru pentru programare precum seturile de mobilier gata făcute pentru construirea de locuințe — este mult mai ușor să iei panouri deja tăiate și să le înșurubezi împreună pentru a face un raft de bibliotecă decât de a realiza singur designul, du-te și găsește lemnul corect, tăie toate panourile la dimensiunea și forma corectă, găsește șuruburile de dimensiunea corectă și apoi le așezi împreună pentru a face o bibliotecă.



APIurile browserului sunt construite în browserul tău web, și pot expune date din mediul înconjurător al calculatoului sau pot face lucruri complexe. De exemplu:

- APIul DOM(Document Object Model) îți pemite să manipulezi codul HTML și CSS, să creezi, să elimini și să modifici codul HTML, să aplici dinamic noi stiluri pe pagina ta, etc. De fiecare dată când vezi o fereastră popup care apare pe o pagină, sau un nou conținut afișat (așa cum am vazut mai sus în demoul nostru simplu) de exemplu, acesta este DOM în acțiune.

- APIul Geolocation preia informațiile geografice. Acesta este modul în care Google Maps poate să îți identifice locația și să te poziționeze pe hartă.
- APIurile Canvas și WebGL îți permit să creezi grafice animate 2D și 3D. Oamenii fac lucruri extraordinare utilizând aceste tehnologii web — vezi Experimente Chrome și webglsamples.
- APIurle Audio și Video precum HTMLMediaElement și WebRTC îți permit să faci lucruri interesante cu multimedia, cum ar fi să redai conținut audio și video chiar în pagina web sau să preiei conținut video de pe camera web și să-l afișezi pe calculatorul altei persoane (încearcă demoul nostru Snapshot să îți faci o idee).

3. HTML

Pentru realizarea (spre exemplu) a container-ului care contine prezentarea mea scurta si poza, secventa HTML arata cam asa :

```
div class="photoContainer">
          <div class="photo"></div>
       </div>
       <div class="aboutMe">
          <div class="textContainer" id="textContainerScroll">
             <br>
             <br>
             <h2 style="text-align: center; font-weight: 700;"
id="aboutAnchor">About Me</h2>
             <br>
             <br>
             <h6 style="text-align: center;">

                I'm Razvan-Antonio Berbece and I'm 19 years old. I was born
in a small city in Romania and I aspire to become my best version of
myself through hard learning, practice and self-motivation.

                <br><br>In the future I want to make one of my wishes
become reality and to be the CEO of my own company which (most
probably) will encourage traveling all across the world through a web-
app. In order to get there, I plan on completing as many internships as
possible and to be up to date with every new and/or useful technology.
```

```
                <br><br>I'm a soon-to-be junior full-stack developer. I provide solutions to front-end and back-end problems, develop small-scaled projects and I'm ready to tackle every challenge which is thrown at me.

                <br><br> I like to learn from my mistakes and I'm always open to new technologies, frameworks and languages due to my desire to be the best version of myself through constant learning and practice.

                <br><br> One of my biggest ambitions is to be able to get to the point in which I will be able to find efficient solutions to whatever problem I encounter. This is why, at the moment I am constantly solving simple algorithms so I can develop this problem-solving skill of mine.

                <br><br> I am a future University of Leeds undergrad, studying Computer Science with an integrated Masters.

            </h6>
        </div>
```

Elementele prezinta clase si id-uri, deoarece in urmatoarea parte a documentatiei vom analiza fisierul CSS, care stilizeaza aceste elemente.

! IMPORTANT : AM FOLOSIT BOOTSTRAP SI MEDIA QUERIES PENTRU DIFERITE TIPURI DE REZOLUTII PENTRU REALIZAREA PAGINII, IN ASA FEL INCAT SA FIE RESPONSIVE (CALITATEA TEXTULUI, INCADRARII SI A IMAGINILOR SA FIE CEA MAI BUNA) PE ORICE TIP DE REZOLUTIE.

4. CSS

Stilizarea elementelor prezentate mai sus s-a facut printr-un fisier extern de CSS. Secventele care stilizeaza acele elemente :

```
.aboutMe {
    overflow: visible;
    box-shadow: 5px -2px 11px 1px rgba(195, 192, 192, 0.92);
    border-radius: 8px;
    margin-top: 65px;
    height: 40em;
    border: 2px solid #737373;
    width: 60%;
    position: absolute;
    left: 20%;
    background-color: rgba(255, 255, 255, 1);
}
.textContainer {
    width: 100%;
    overflow: auto;
    height: 100%;
}
#textContainerScroll::-webkit-scrollbar {
    width: 0.5em;
    height: 0.5em;
}
#textContainerScroll::-webkit-scrollbar-thumb {
```

```
    background-color: rgb(218, 218, 218);
    border-radius: 100px;
}
#textContainerScroll::hover {
    background: rgb(189, 189, 189);
}
.content-wrapper {
    position: relative;
    height: 100vh;
    margin-top: 100px;
}
.photo {
    text-align: center;
    width: 100px;
    height: 100px;
    background-image: url("images/eu.jpg");
    border-radius: 100%;
    box-shadow: 3px 3px 7px 3px rgba(195, 192, 192, 0.92);
    background-size: contain;
}
.photoContainer {
    position: absolute;
    z-index: 9999;
    justify-content: center;
    display: flex;
    width: 100%;
}
```

Dupa cum se poate observa :

- Se pune accent pe folosirea unitatilor relative, tocmai pentru a promova caracterul responsive
- Se folosesc path-uri relative&locale pentru imagini => eficienta la incarcarea paginii
- Acel “-webkit-scrollbar” intervine doar atunci cand pe o anumita rezolutie, textul din elemental .AboutMe depaseste limitele containerului in care este pus. In acest caz, va aparea o bara de scroll stilizata, la marginea containerului.
- Se folosesc pozitii relative pentru elementele mari si absolute pentru cele ce apartin elementelor majore.

```
@media (max-width: 770px) and (orientation: landscape) {
  .tech1,
  .tech2 {
    border-right: none !important;
  }
  .projectsContainer {
    height: 1040px;
  }
  .buttonContainer {
    margin-top: 63%;
  }
  .projectBtn {
    height: 55px;
    width: 15%;
  }
  .contactContainer {
    width: 70%;
    left: 15.5%;
```

```
    }
    .contactText {
        font-size: 18px;
    }
    .contactIcon {
        font-size: 3em;
    }
    .galleryContainer {
        left: 8%;
        width: 85%;
    }
    .galleryImage {
        padding-bottom: 5px;
    }
    .photoRow {
        margin-left: 12%;
    }
    .BTC2 {
        margin-top: 1.3% !important;
    }
    .BT2 {
        width: 65px;
        height: 35px;
    }
}
```

Cum am spus mai sus, am folosit media queries pentru un comportament responsive. Un exemplu este reprezentat de

comportamentul unor elemente pe o rezolutie de telefon, aflat in modul Landscape este cea din secventa de sus (evidentiata cu rosu).

5.Javascript & Jquery

Pentru realizarea unora dintre animatii sau pentru schimbarea proiectelor in sectiunea aferenta s-a folosit limbajul de programare Javascript si, respective, Jquery (desi nu mai primeste update-uri, acest “framework” ursureaza mult treaba unor programatori).

Spre exemplu, cand butonul “Hide” sau “Show” de pe bara din footer-ul paginii este apasat, are loc o animatie. Codul sursa al animatiei este :

```
var MediaFooter = $('.MediaFooter');
   var Hide = $('.HideButton');
   var Show = $('.ShowButton');
   var M1 = $('.media1');
   var M2 = $('.media2');
   var M3 = $('.media3');
   Hide.on('click', HideBar);
   Show.on('click', ShowBar)

   function HideBar() {
      MediaFooter.animate({
         width: "115px",
```

```
        borderRadius: "21px",
        marginLeft: "15px",
        marginBottom: "15px"
    }, 1500, 'linear');
    M1.fadeOut(1000);
    M2.fadeOut(1000);
    M3.fadeOut(1000);
    Hide.fadeOut(500);
    Show.fadeIn(1000);
}

function ShowBar() {
    MediaFooter.animate({
        width: "100%",
        borderRadius: "0",
        marginLeft: "0",
        marginBottom: "0"
    }, 1000, 'linear');
    M1.fadeIn(1000);
    M2.fadeIn(1000);
    M3.fadeIn(1000);
    Hide.fadeIn(500);
    Show.fadeOut(1000);
}
```

Se poate observa faptul ca schimbam atribute din CSS folosing functia .animate.

Hide primeste adresa butonului care ascunde bara de retele de socializare, in timp ce Show primeste adresa butonului ce extinde bara dup ace a fost ascunsa. Astfel, prin sintaxa (variabila).on(‘click’, functie()), realizam acea animatie.

6. Cerinte Hardware

CPU: Intel Pentium 4 AMD Athlon

RAM: 1GB RAM

Plăcă video: 256MB

Spaţiu hard-disk: 50 MB

Rezolutie ecran 800x600 1024x768

Cerintele nu sunt deloc mari, avand in vedere faptul ca un portofoliu digital se bazeaza pe faptul ca atrage angajatori prin simplicitatea si totusi aspectul lui modern. Prin urmare, nu au fost folosite animatii complicate, sute de poze care sa ingreuneze procesul de incarcare a datelor sau alte lucruri de acest tip.

7. Manual de Utilizare

a. Prima bara de navigatie:
   - Butonul “Documentation” va deschide o pagina noua care continue documentatia proiectului in format PDF.
   - Butonul “Gallery” va muta bara de scroll a site-ului la sectiunea “Gallery” a paginii.
   - Butonul “CV” va deschide o pagina noua care continue CV-ul meu in format PDF.

b. A doua bara de navigatie:
   - Butonul “Home” actioneaza ca o revenire in varful paginii.
   - Butonul “Projects” va muta bara de scroll la sectiunea “Projects” care continue proiectele la care am lucrat pana in acest moment, plus o descriere a site-ului, a tehnologiilor folosite si a problemelor intalnite (si a modului de rezolvare a acestora). Dand click pe poza unui proiect, veti fi redirectionat la proiectul respectiv.
   - Butonul “Contact” va muta bara de scroll la sectiunea de contact, unde se vor gasi datele mele de contact.

c. Bara fixata in footer-ul paginii continue link-uri catre diverse medii de socializare sau de contact:
    - Github
    - Facebook
    - Instagram

8. Bibliografie

Pentru realizarea acestei documentatii, am folosit urmatoarele surse de informare :

https://azimutvision.ro/unit/introducere-in-css/

https://developer.mozilla.org/ro/docs/Learn/Getting_started_with_the_web/HTML_basics

https://storeday.ro/tutorial-despre-html5-css3-si-javascript-web-design/

De asemenea:

- **Pozele au fost realizate de mine.**
- **Iconitele prezente pe pagina fac parte din CDN-ul Font Awesome.**

9. Concluzii

In concluzie, am aprofundat multe dintre cunostintele mele ce tin de web developing realizand acest proiect si toate celelalte prezentate pe site. In plus, faptul ca am realizat un portofoliu digital imi ofera o potentiala baza (ce trebuie imbunatatita) pentru un viitor interviu de angajare.

//SFARSIT DOCUMENTATIE