IZRADA SUSTAVA ELEKTRONIČKOG UČENJA IZ ROBOTIKE

Popović, Josip

IZRADA SUSTAVA ELEKTRONIČKOG UČENJA IZ ROBOTIKE

Popović, Josip

Veleučilište u Karlovcu
Strojarski odjel
Stručni studij mehatronike

Josip Popović

IZRADA SUSTAVA ELEKTRONIČKOG UČENJA IZ ROBOTIKE

ZAVRŠNI RAD

Karlovac, 2020. godina

Karlovac University of Applied Science
Mechanical Engineering Department
Professional undergraduate study of Mechatronics

Josip Popović

DEVELOPMENT OF THE E-LEARNING SYSTEM FOR ROBOTICS

Final paper

Karlovac, 2020.

Veleučilište u Karlovcu
Strojarski odjel
Stručni studij mehatronike

Josip Popović

IZRADA SUSTAVA ELEKTRONIČKOG UČENJA IZ ROBOTIKE

ZAVRŠNI RAD

Mentor: dr. sc. Adam Stančić, v. pred.

Karlovac, 2020.

SAŽETAK

Robotika je grana inženjerske znanosti te ima cilj napraviti stroj koji bi pomagao ljudima u svakodnevnim poslovima. Obuhvaća razna znanstvena područja poput strojarstva, elektronike, računarstva i automatike stoga je korisno izraditi sustav elektroničkog učenja koji bi pomogao objediniti većinu područja.

Elektroničko učenje je način učenja pomoću elektroničkih uređaja koji su povezani na Internet s ciljem da poboljša efikasnost i kvalitetu učenja. Kako tehnologija sve brže napreduje, ovakav način učenja postaje sve više dostupniji. Za učinkovit i uspješan sustav elektroničkog učenja potreban je dobar dizajn kako bi lakše privukao korisnike te kvalitetnije prenio znanje. U ovom radu će biti prikazana izrada web aplikacije kao sustava električnog učenja iz robotike. Web aplikacija je grafičko sučelje kojim se pristupa pomoću Internet preglednika. Izrada web stranica ili aplikacija se odnosi na postupak izrade i održavanja web stranice ili aplikacije što se može odnositi na web dizajn, programiranje, upravljanje bazama podataka i postavljanje izrađene web stranice ili aplikacije na web server.

Najčešće se koriste dva načina izrade web stranice ili aplikacije: pomoću CMS sustava, odnosno programa koji pomaže pri izradi u kojem nije potrebno znanje programiranja i ručna izrada. U ovom radu će biti prikazan primjer ručne izrade, odnosno korištenjem skriptnih jezika HTML-a i CSS-a te programskog jezika JavaScript.

Ključne riječi: robotika, elektroničko učenje, web aplikacija, izrada web aplikacije

SUMMARY

Robotics is a branch of engineering science and aims to make a machine that would help people in their daily tasks. It covers various scientific fields such as mechanical engineering, electronics, computer science and automation, so it is useful to create an e-learning system that would help unify most areas.

E-learning is a way of learning using electronic devices connected to the Internet in order to improve the efficiency and quality of learning. As technology advances faster and faster, this way of learning is becoming more and more accessible. An efficient and successful e-learning system requires a good design in order to attract users more easily and transfer knowledge better. In this paper, the development of a web application as an electrical learning system in robotics will be presented. A web application is a graphical interface that is accessed using an Internet browser. Website or application development refers to the job of building and maintaining a website or application which may relate to web design, programming, database management and placing the created website or application on a web server.

There are two ways of creating a website that are most commonly used: using a CMS, a program that helps with the creation of which does not require knowledge of programming and manual development. This paper will show an example of manual creation, ie using the scripting languages HTML and CSS and the programming language JavaScript.

Keywords: robotics, e-learning, web application, web application development

SADRŽAJ

1. UVOD

Mreža svih mreža u nekoliko je desetljeća prekrila veći dio planete i povezala milijarde ljudi. Danas je mnogima od nas život nezamisliv bez Interneta. Kako tehnologija napreduje tako je sve više prisutan u našoj svakodnevici. Ukratko rečeno, Internet je globalna mreža milijardi računala i drugih elektroničkih uređaja. Internet sačinjavaju bezbrojne kućne, državne, poslovne i akademske mreže, koje su međusobno povezane i koje međusobno razmjenjuju informacije. Kako bi se to postiglo postavljeno je na desetke tisuća kilometara kabela između država, otoka i kontinenata. Prema izvoru [1], u pojedinim državama gotovo su svi online, npr. u Islandu je 98 posto ljudi povezano s Internetom, a slično je i u Danskoj. U Velikoj Britaniji je 95 posto ljudi online, u SAD-u 89 posto, u Španjolskoj 85 posto, a u Njemačkoj 84 posto. S druge strane, postoje i mjesta gdje je taj postotak dosta niži, pri čemu je glavni razlog siromaštvo. Puno je mjesta poput Tanzanije, Ugande i Sudana, gdje tek 30 do 40 posto ljudi je online. Ima i država poput Gvineje, Liberije i Sierra Leonea, gdje je tek sedam do 11 posto ljudi online. Struktura i poglavlja završnog rada formirani su na sljedeći način. U drugom poglavlju je detaljnije objašnjen Internet, kako funkcionira, kako je nastao i njegova povijest. U trećem poglavlju je opisano što je to elektroničko učenje te prednosti i nedostaci elektroničkog učenja. U četvrtom poglavlju objašnjeno je što je web dizajn i potrebne vještine koje treba imati svaki web dizajner. U petom poglavlju su pojašnjena dva načina izrade web stranice: s CMS sustavom i ručnom izradom bez pomoćnih aplikacija i sustava. U šestom poglavlju su opisani postupci za izradu web stranice od početka na primjeru gdje je glavna tema web stranice robotika. U posljednjem poglavlju je opisano postavljenje kreirane web stranice na Internet [1].

2. OPĆENITO O INTERNETU

Internet je globalni sustav međusobno povezanih računalnih mreža koji koriste skup pravila za komunikaciju između mreža i uređaja. To je mreža koja se sastoji od privatnih, javnih, akademskih, poslovnih i vladinih mreža povezanih elektroničkim, bežičnim i optičkim tehnologijama [2]. Od njegova začetka do danas unutar njega su razvijeni mnogi servisi od kojih su najpoznatiji World Wide Web, daljinski prijenos podataka i elektronička pošta [1].

2. 1. Povijest razvoja Interneta

4. listopada 1957. godine, istog dana kad je započela svemirska utrka na kozmodromu Baikonur u Kazahstanu, lansiranjem sovjetskog satelita Sputnik je pokrenut pokus ARPANET (engl. Advanced Research Projects Agency Network). Za vrijeme hladnog rata Sputnik je predstavljao prijetnju zapadu i veliko znanstveno postignuće u odnosu na Ameriku. Kako bi sustigli Ruse, američki predsjednik Dwight Eisenhower dodijelio je ARPA-i (engl. Advanced Research Project Agency) sve programe od R&D (engl. Defence Research and Development). Unatoč nekim uspjesima od ARPA-e, njihovo sudjelovanje u svemirskoj utrci nije dugo trajalo pošto je američka vlada odlučila osnovati novu agenciju, NASA (engl. National Aeronautics and Space Administration) kako bi se učinkovitije nosili s političkim pritiskom. Kako ARPA ne bi pala u zaborav, svoje rješenje je pronašla u računalnim znanostima pod vodstvom Josepha Carla R. Licklidera. U svom seminarskom radu Čovjek-računalna simbioza (engl. Man–Computer Symbiosis) iz 1960. godine, Licklider je napisao „*human brains and computing machines will be coupled together very tightly*" [3].

1962. godine, ARPA-in odjel za istraživanje i kontrolu zapovjedništva je postao IPTO (engl. Information Processing Techniques Office) pod vodstvom J.C.R. Licklidera, a kasnije pod Ivanom Sutherlandom. Glavna funkcija IPTO-a je bila odabir, financiranje i koordinacija istraživačkih projekata sa sjedištem u SAD-u koji su se fokusirali na napredne i mrežne tehnologije. Do kraja 1960-ih, analogni sklopovi telefonske mreže nisu bili pouzdani pošto nije

bilo neuobičajeno da se čitavi skupovi informacija izgube na putu od terminalnog računala do glavnog računala [4].

RAND (engl. Research and Development) korporacija je osnovana 1946. godine sa sjedištem u Santa Monici u Kaliforniji te je dan danas neprofitna institucija koja pruža istraživanje i analize u širokom rasponu polja. Paul Baran, jedan od ključnih istraživača RAND-a je 1964. godine objavio rad pod naslovom „Raspodijeljena komunikacija" u kojem je istaknuo komunikacijski sustav koji bi mogao izdržati nuklearni napad. Baranova idealna mreža je bila ispred svog vremena pošto se takva mreža jedino može održati u potpunom razvijenom digitalnom okruženju. Zamislio je da će se par manjih računala koristiti kao usmjerivači, ali ta tehnologija tada nije bila dostupna. RAND je vjerovao u Baranov projekt ali nisu uspjeli pronaći partnere s kojima bi mogli surađivati. 1965. godine su bili prvo odbijeni od strane zrakoplovnih snaga, a zatim od AT&T (engl. American Telephone and Telegraph) [4].

Slika 1: Paul Baran [4]

Charles Herzfeld, direktor ARPA-e je dao dopuštenje za početak prve faze projekta ARPANET te Robert Taylor je počeo predlagati plan nekim ARPA-inim poslovnim suradnicima. Slično kao i Baran, Taylor je ubrzo shvatio da dobre ideje nije lako prodati. Projekt je zaživio kada je Lawrence G. Roberts, talentiran istraživač iz Lincolnovog laboratorija na MIT-u (engl. Massachusetts Institute of Technology) izabran za voditelja projekta ARPANET. 1966. godine,

Roberts sa svojim kolegom Thomasom Marrillom je koristio telefonsku liniju kompanije Western Union kako bi povezao dva super računala (Q32 u korporaciji za razvoj sustava (engl. System Development Corporation) u Santa Monici, Kalifornija i TX-2 u laboratoriju Lincoln u Lexington, Massachusetts) s ciljem da provjeri je li moguće izgraditi računalnu mrežu na kontinentalnom opsegu. Roberts i Marrill su dokazali da je moguće povezati dva različita računala i dijeliti podatke između njih te kako bi poboljšali brzinu i pouzdanost mreže morali su koristiti metodu komutacija paketa (engl. packet switching) [5].

Roberts je počeo skicirati planove za mrežu te nakon rasprave o mrežnim specifikacijama s kolegama je smislio dvije neophodne značajke za mrežu: protokol koji je svih 16 istraživačkih skupina moglo prihvatiti i kapacitetom da podrži procijenjenih 500 000 paketa prometa dnevno između 35 računala koja su bila povezana sa 16 domaćina (engl. hosts). Prema Baranovoj izvornoj ideji, ARPANET je postala raspodijeljena mreža koja je koristila usmjerivače, odnosno IMP (engl. Interface Message Processors) na svakom čvoru kako bi ubrzala komunikaciju između računala. 29. srpnja 1968. godine ARPA je izdala RFQ (engl. Request for Quotation), odnosno poziv za podnošenje financijskih ponuda nekoliko tvrtki u računalnom sektoru radi izgradnje IMP-ova. IBM (engl. International Business Machines) i CDC (engl. Control Data Corporation) su odbili ponudu jer nisu vjerovali da metoda komutacija paketa može uspjeti. Dva najbolja kandidata za ugovor su bili BBN (engl. Bolt, Beranek i Newman) i Raytheon. Prema Robertsovom iskustvu, Raytheonova birokratska struktura bi zakomplicirala stvari i usporila bi cijeli projekt zbog toga što bi se izgubilo više vremena na pronalazak prave osobe s kojom bi se moglo razgovarati o projektu. U siječnju 1969. godine BBN je dobio ugovor od milijun dolara za izgradnju četiri IMP-a za mrežu s četiri mjesta do kraja te godine što je bio jasan znak anti-birokratske prirode oko koje je izgrađen Internet [5].

Prva četiri čvora mreže ARPANET bili su kalifornijsko sveučilište Los Angeles (UCLA engl. University of California Los Angeles), kalifornijsko sveučilište Santa Barbara (UCSB engl. University of California Santa Barbabra), sveučilište u Utahu i istraživački institut Stanford (SRI engl. Stanford Research Institute). 1. rujna 1969. godine je instalirano prvo računalo u UCLA te je izabrano zbog Leonarda Kleinrocka i njegovog mrežnog centra za mjerenje koje je financirala APRA. Centar se fokusirao na analizu i mjerenje mrežnog prometa te iznošenje statistika koje će se koristiti u provedbi mreže. 29. listopada 1969. godine je ostvarena prva veza između UCLA i

SRI zbog Kleinrockovog i Charley Klineovog pokusa te je to bio prijenos poruke između računala UCLA SDS Sigma 7 i računala SRI SDS 940 [5]. Prvu poruku preko ARPANET-a je poslao Kline gdje je uspio samo poslati slova „l" i „o" iz riječi „login". Poruka je označila važnu prekretnicu jer je ubrzo promijenila način na koji ne samo strojevi, već i ljudi komuniciraju [3].

Početkom 70-ih kada su prvi čvorovi ARPANET-a postali potpuno funkcionalni, razmjena podataka između računala nije bila jednostavna. Postojala je pouzdana mreža na koju se moglo povezati, ali ne i osnovna pravila za komunikaciju pošto je svaki domaćin ima svoj skup pravila i vlastiti jezik koji su korisnici morali znati. Kako bi se riješio taj problem, Roberts i njegovo osoblje osnovali su grupu istraživača koja se zvala NWG (engl. Network Working Group). Grupu je vodio Steve Crocker, diplomski student UCLA-a te su imali cilj da razviju softver domaćin-domaćin (engl. host-to-host). Komunikacija u grupi se odvijala slanjem zahtjeva za komentar (RFC engl. Request for Comment) te je takav način komunikacije pomogao članovima grupe da otvoreno razmišljaju o ciljevima mreže [6].

NWG-u je bilo potrebno dvije godine da napiše softver, ali je ipak 1970. godine ARPANET imao svoj protokol NCP (engl. Network Control Protocol). Korisnici na domaćim računalima su bili uspješno priključeni na mrežu, ali im je nedostajala ispravna konfiguracija ili znanje za pravilno korištenje mreže. Kako bi se riješio taj problem, 1972. godine Roberts i njegove kolege su javno održali demonstraciju o ARPANET-u i njegovim potencijalima. Demonstracija je u potpunosti uspjela pokazati korisnicima koji nisu bili uključeni u izgradnju projekta kako metoda komutacije paketa funkcionira te je inspirirala druge da slijede primjer koji je postavila Robertsova mreža. 1972. godine uveden je koncept „Internetting", odnosno „mreže otvorene arhitekture" koji je ilustrirao kritičnu potrebu da se mreža proširi izvan svog ograničenog kruga. Protokol NCP nije ispunio zahtjev pošto je dizajniran za upravljanje komunikacijom unutar iste mreže [6].

Za izgradnju pouzdane i dinamičke mreže potreban je bio novi protokol. 1978. godine, Robert Kahn i Vint Cerf, dvojica iz BBN-a su uspjeli dizajnirati TCP/IP (engl. Transfer Control Protocol/Internet Protocol) protokol koji je otvorio nove mogućnosti za suradnju s ARPANET-om i ostalim mrežama širom svijeta te su postavljeni temelji svjetske mreže. U mreži je već bilo 57 čvorova što govori da je sve teže bilo odrediti tko zapravo koristi mrežu te bi to moglo postati ozbiljan problem za nacionalnu sigurnost. 1. srpnja 1975. godine ARPANET mreža je stavljena

pod izravnu kontrolu DCA (engl. Defense Communication Agency). Pokušavajući kontrolirati situaciju, DCA je izdala niz upozorenja protiv neovlaštenog pristupa i upotrebe mreže ali su uglavnom zanemarena. Ovu situaciju je još pogoršao drastičan pad cijena računala. ARPANET je bio podijeljen u dvije različite mreže: ARPANET, uglavnom posvećena istraživanju i MILNET (engl. Military Network), vojna operativna mreža [6].

Sredinom 1980-ih mrežu su koristili istraživači i programeri ali su je također počeli koristiti druge zajednice i mreže. Prelazak na privatizirani Internet je trajao još deset godina gdje je najveći utjecaj imao NSF (engl. National Science Foundation). NSF-ova vlastita mreža NFTNET je koristila ARPANET kao temelje za svoju mrežu od 1984. godine. 1989. godine Tim Berners-Lee je izumio WWW (engl. World Wide Web). ARPANET je službeno povučen 1990. godine, dok je 1995. godine NFTNET bio ugašen te Internet privatiziran. Tijekom 1992. godine, globalne mreže povezane s Internetom razmijenile su oko 100 gigabajta prometa dnevno te je od tada podatkovni promet eksponencijalno rastao, zajedno s brojem korisnika i popularnošću mreže. 2014. godine, globalni Internetski promet dosegao je 16 000 Gbps [6].

2. 2. Osnovni pojmovi Interneta

Poslužitelj (engl. server) je softverski ili hardverski uređaj koji prihvaća i odgovara na zahtjeve upućene preko mreže. Serveri se koriste za upravljanje mrežnim resursima. Na primjer, korisnik može postaviti server za kontrolu pristupa mreži, slanje / primanje elektroničke pošte, upravljanje zadacima ispisa ili za postavljanje web stranice [7]. Korisnik (engl. client) je računalo koje se povezuje i koristi podatke udaljenog računala ili servera [8].

Protokol znači sustav pravila i propisa koji nešto upravljaju, odnosno neka vrsta „službenog postupka" ili službenog načina na koji se nešto mora učiniti. Postoje protokoli za razne namjene, stoga različiti web servisi upotrebljavaju različite protokole [9]. IP protokol je skup pravila za usmjeravanje, fragmentaciju i adresiranje paketa podataka kako bi mogli putovati mrežom i stići na točno odredište. Paketi su podaci koji su podijeljeni u manje dijelove. IP informacije priložene su svakom paketu te pomažu usmjerivačima slanje paketa na pravo odredište [10]. IP adresa je brojčana oznaka dodijeljena svakom uređaju koji je spojen na mrežu

koja koristi IP protokol te je formirana od četiri troznamenkaste vrijednosti između 0 i 255. IP adresa ima dvije glavne funkcije: identifikacija mrežnog sučelja ili lokalno adresiranje [11]. Privatna IP adresa je IP adresa koje se koristi za komunikaciju unutar lokalne računalne mreže te korištenjem se može slati ili primati podatke unutar iste mreže. Javna IP adresa je IP adresa koja se koristi za komunikaciju izvan lokalne mreže te ju u osnovi dodjeljuje ISP (engl. Internet Service Provider), odnosno tvrtka koja pruža pristup Internetu [12].

TCP (engl. Transmission Control Protocol) je transportni protokol koji određuje način slanja i primanja podataka. TCP zaglavlje je uključeno u podatkovni dio svakog paketa koji koristi TCP/IP protokol. Prioritet TCP je pouzdanost budući da mora osigurati da svi paketi dođu po redu, stoga učitavanje podataka može trajati duže ako neki paketi nedostaju. UDP (engl. User Datagram Protocol) je transportni protokol koji je brži od TCP-a ali je manje pouzdan pošto ne osigurava isporuku i redoslijed svih paketa. UDP/IP se obično koristi za streaming audio ili video zapisa jer su to slučajevi u kojima je bitnije održavanje prijenosa u stvarnom vremenu nego mogući odbačeni paketi [10].

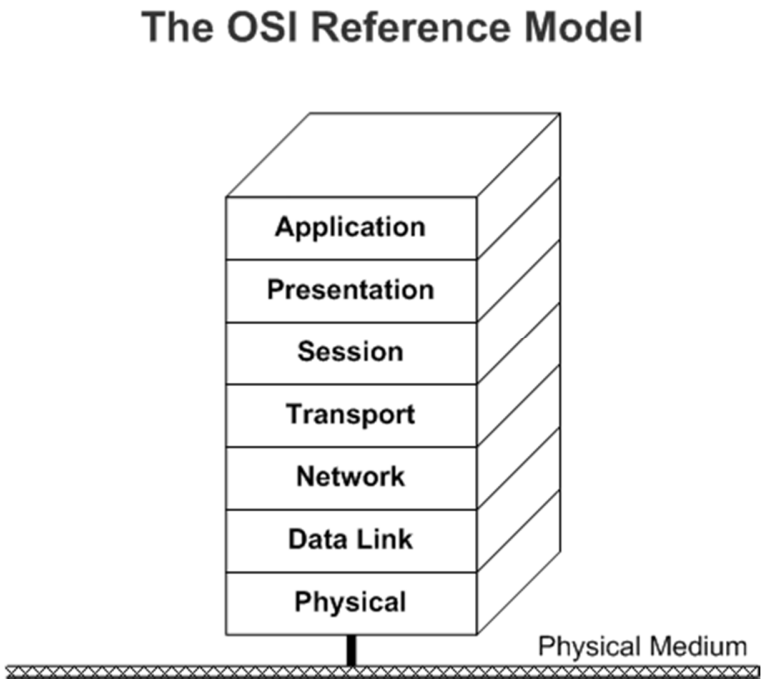
Domena (engl. domain) je oznaka koja se koristi umjesto IP adrese. Na primjer, ime domene „google.com" upućuje na IP adresu „216.58.216.164". Generalno, lakše je zapamtiti ime, a ne dugačak niz brojeva. Naziv domene može biti najviše šezdeset tri znaka [13]. DNS (engl. Domain Name System) je usluga koja prima zahtjev koji sadrži ime domaćina domene i odgovara s odgovarajućom IP adresom. Istim imenom se naziva i računalo, server, na kojem se taj program izvodi. Kad korisnik želi pristupiti stranicama kompanije Google, može upisati „https://www.google.com" u adresnu traku svog preglednika. Nakon unosa naziva domene, potražit će se u sustavu imena domena gdje je preveden u IP. Pomoću IP adrese računalo može tada pronaći web stranicu Google i proslijediti te podatke pregledniku [14].

Kolačići (engl. cookies) su datoteke koje se razmjenjuju između korisnikovog računala i web servera radi prepoznavanja određenih korisnika i poboljšanja njihovog iskustva pregledavanja [15]. Svrha kolačića je pomoći vlasnicima i administratorima web stranica da prate posjete i aktivnosti. Na primjer, mnogi mrežni trgovci koriste kolačiće za praćenje predmeta u korisničkoj košarici dok korisnici istražuju web stranicu. Bez kolačića, sadržaj web košarice bi se resetirao na nulu svaki put kad se klikne na novi link na web stranici [16].

2. 3. OSI model

OSI (engl. Open Systems Interconnection) model je konceptualni model koji je kreirala Međunarodna organizacija za standardizaciju te koji omogućuje raznim komunikacijskim sustavima komunikaciju koristeći standardne protokole. Iako moderni Internet ne slijedi strogo OSI model, još uvijek je vrlo koristan za rješavanje mrežnih problema. Bez obzira da li se jedna osoba ne može spojiti na Internet ili web stranica koja ima na tisuće korisnika je prestala funkcionirati, OSI model može pomoći u rješavanju problema [17].

OSI model dijeli funkcije, protokole i mrežne uređaje u različite slojeve. Neke od prednosti slojevitog pristupa su: komunikacija je podijeljena na manje i jednostavnije komponente, promjene u jednom sloju ne utječe na ostale slojeve, lakše je standardizirati funkcije kada su podijeljene na manje dijelove te omogućuje besprijekornu međusobnu komunikaciju različitih vrsta hardvera i softvera. OSI model se sastoji od sedam slojeva koji su prikazani na slici ispod [18].



Slika 2: OSI model [19]

Aplikacijski sloj pruža sučelje između aplikacije na sustavu te je jedini sloj koji izravno komunicira s podacima korisnika. Aplikacije poput web preglednika i elektroničke pošte oslanjaju se na aplikacijski sloj za pokretanje komunikacije. Ovaj sloj je odgovoran za protokole i manipulaciju podacima na koje se softver oslanja kako bi korisniku prikazao korisne podatke te aplikacije klijentskog softvera nisu dio aplikacijskog sloja [17].

Prezentacijski sloj je prvenstveno odgovoran za pripremu podataka tako da ga aplikacijski sloj može koristiti te je odgovoran za prijevod, šifriranje i kompresiju podataka. Dva komunikacijska uređaja mogu koristiti različite metode kodiranja stoga je ovaj sloj odgovoran za prevođenje dolaznih podataka u sintaksu koju aplikacijski sloj može razumjeti. Ako uređaji komuniciraju preko šifrirane veze, ovaj sloj je odgovoran za dodavanje enkripcije kao i za dekodiranje na strani primatelja tako da može aplikacijskom sloju poslati čitljive podatke. Ovaj sloj je također odgovoran za kompresiju podataka koje dobije iz aplikacijskog sloja prije nego što ih pošalje prema sloju sesije [17].

Sloj sesije je odgovorna za otvaranje i zatvaranje komunikacije između dva uređaja, odnosno osigurava da sesija ostane dovoljno otvorena za prijenos svih podataka, a zatim brzo zatvara sesiju kako se ne bi trošili resursi. Također sinkronizira prijenos podataka s kontrolnim točkama. Za prenošenje datoteka od 50 megabajta, sloj sesije može postaviti kontrolnu točku na svakih 5 megabajta. Ako dođe do prekida veze nakon prenesenih 27 megabajta, sesija se može nastaviti s posljednje kontrolne točke što znači da će biti potrebno prenijeti još samo 25 megabajta [17].

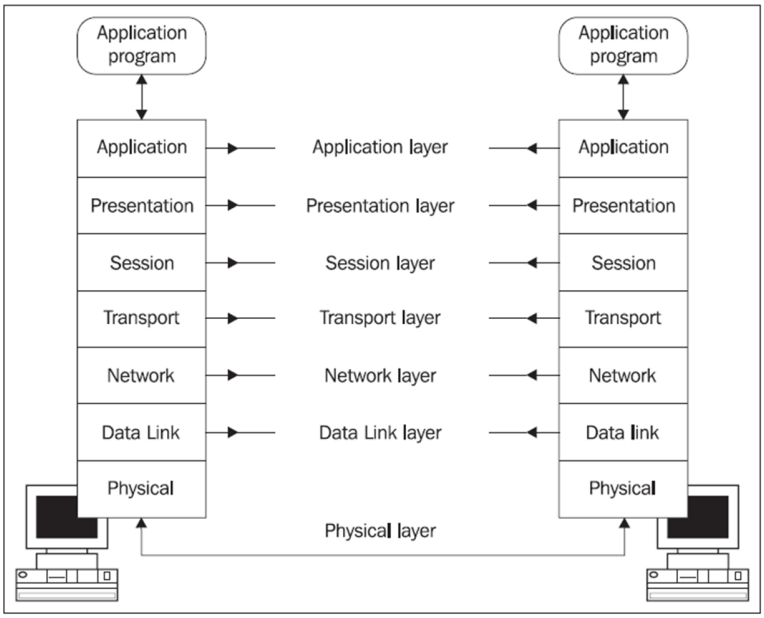
Transportni sloj je odgovoran za međusobnu komunikaciju između dva uređaja. Uzima podatke iz sloja sesije i razdjeljuje u manje dijelove podataka koji se nazivaju segmenti prije slanja u mrežni sloj te na prijemnom uređaju sastavlja segmente u podatke koje onda šalje prema sloju sesije. Ovaj sloj također ima funkciju kontrole protoka i pogrešaka. Kontrola protoka određuje optimalnu brzinu prijenosa kako bi se osiguralo da pošiljatelj brzom vezom ne optereti prijemnik sa sporom vezom. Kontrola pogreške se vrši na kraju prijema osiguravajući da su primljeni podaci potpuni ili zahtijevajući ponovni prijenos ako podaci nedostaju [17].

Mrežni sloj je odgovoran za olakšavanje prijenosa podataka između dvije različite mreže te ako uređaji komuniciraju na istoj mreži, tada mrežni sloj nije potreban. Na strani pošiljatelja

ovaj sloj rastavlja segmente transportnog sloja u manje dijelove podataka koji se nazivaju paketi, a na prijemnom uređaju ih sastavlja. Također pronalazi najbolji fizičku putanju da podaci stignu na svoje odredište [17].

Sloj veze olakšava prijenos podataka između dva uređaja na istoj mreži. Uzima podatke s mrežnog sloja i rastavlja na manje dijelove podataka koji se zovu okviri (engl. frames) [17]. Ovaj sloj definira protokole koji se koriste za slanje i primanje podataka te otkriva koliziju i druge pogreške u primljenim podacima [18].

Fizički sloj uključuje fizičku opremu koja je uključena u prijenos podataka, poput kablova i preklopnika. Ovaj sloj na oba uređaja se moraju dogovoriti oko konvencije signala te se podaci pretvaraju u binarni kod, odnosno 0 i 1 [17].



Slika 3: Komunikacija između dva računala pomoću OSI modela [20]

Princip rada OSI modela će biti objašnjen na primjeru pomoću gornje slike. Korisnik A želi poslati email korisniku B. Aplikacija emaila korisnika A proslijedit će svoju poruku emaila na aplikacijski sloj, koji će koristiti protokol SMTP (engl. Simple Mail Transfer Protocol) te proslijediti podatke prezentacijskom sloju koji će komprimirati podatke. Podaci putuju prema sloju sesije koji će inicijalizirati sesiju komunikacije te poslati podatke prema transportnom sloju gdje će biti segmentirani. Na mrežnom sloju segmenti će biti podijeljeni u pakete koji će se još

podijeliti u okvire na sloju veze. Okviri će biti dostavljeni u fizički sloj koji će podatke pretvoriti u binarni kod i poslati ih preko fizičkog medija. Ovaj cijeli proces se naziva enkapsulacija [17].

Nakon što računalo korisnika B primi podatke u binarnom kodu putem fizičkog medija, podaci će proći kroz iste slojeve jedino obrnutim redoslijedom. Fizički sloj će pretvoriti binarni kod u okvire koje će poslati prema sloju veze. Sloj veze će sastaviti okvire u pakete te ih poslati prema mrežnom sloju. Mrežni sloj će od paketa napraviti segmente te ih sastaviti u podatke. Nakon toga sloj sesije će proslijediti podatke prezentacijskom sloju te završiti sesiju komunikacije. Prezentacijski sloj će ukloniti kompresiju te proslijediti podatke aplikacijskom sloju koji će pomoću aplikacije emaila korisniku B prikazati poruku na zaslonu. Ovaj cijeli proces se naziva dekapsulacija [18].

2. 4. Razmjena podataka na Internetu

Način na koji Internet funkcionira i kako podaci putuju će biti objašnjen na primjeru gdje korisnik želi pročitati članak. Korisnik prvo otvori web preglednik i poveže se s web stranicom koju je posjetio. Računalo korisnika šalje elektronički zahtjev putem internetske veze prema DNS-u. DNS traži podudaranje između imena domene i IP adrese gdje ako pronađe usmjerit će zahtjev na odgovarajuću IP adresu servera [21].

Kada zahtjev stigne do web servera tražene web stranice, server će odgovoriti slanjem tražene datoteke u nizu paketa. Paketi su dijelovi datoteka veličine između 1000 i 1500 bajta. Također sadrže zaglavlja i podnožja koji govore računalima što se nalazi u paketu i kako se informacije uklapaju u ostale pakete kako bi kreirali cijelu datoteku. Svi paketi ne moraju nužno ići istim putem pošto obično prolaze optimalnom rutom. Kada paketi stignu na odredište, uređaj korisnika ih slaže prema određenom protokolu te se rezultat prikaže kao članak na web stranici [21].

3. ELEKTRONIČKO UČENJE

Prije pojave Interneta, model obrazovanja je bio prilično jasan. Prisutnost nastavi je bio jedini način izvođenja nastave, dok je bilo koji drugi način učenja bio upitan. Elektroničko učenje (engl. e-learning) je industrija koja se brzo razvija u sustav budućeg formalnog i neformalnog učenja [22]. Elektroničko učenje je učenje i obuka putem digitalnih izvora i pruža se putem elektroničkih uređaja poput računala, tableta i mobitela koji su povezani s Internetom. Tako se korisnicima olakšava učenje bilo kada i bilo gdje [23].

3. 1. Povijest elektroničkog učenja

Da bi se bolje razumjelo kako elektroničko učenje pridonosi organizacijama danas, korisno je pogledati njegovu prošlost. Elliott Maisie skovao je termin „elektroničko učenje" 1999. godine [24]. Davno prije nego što je započeo Internet, tečajevi na daljinu nudili su se korisnicima za obrazovanje o određenim predmetima ili vještinama. U 1840-ima Isaac Pitman učio je svoje učenike kratkim rukopisom putem prepiske. Ovaj oblik simboličkog pisanja osmišljen je kako bi se poboljšala brzina pisanja i bio je popularan među tajnicima, novinarima i drugim pojedincima koji su mnogo bilježili ili pisali [25].

1924. izumljen je prvi stroj za testiranje i omogućavao je korisnicima da se sami testiraju. 1954., BF Skinner, profesor s Harvarda, je izumio „nastavni stroj" koji je obrazovnim institucijama omogućio da upravljaju programiranim podučavanjem svojih korisnika. Svijetu je predstavljen prvi program za računalnu nastavu 1960. godine. Ovaj program temeljen na računalu bio je poznat kao PLATO-programirana logika za automatizirane nastavne operacije. Prvi sustavi internetskog učenja postavljeni su samo za pružanje informacija učenicima, ali krajem 20. stoljeća, uvođenjem računala i Interneta su se proširili alati za elektroničko učenje i metode isporuke. Računala kompanija Apple, Macintosh je omogućio korisnicima korištenje računala u svojim domovima te tako olakšao im učenje i razvijanje određenih vještina. [25].

3. 2. Razlika između učenja i obuke

Dobar obrazovni proces mora se sastojati i od učenja i obuke, stoga je važno razumjeti razliku između učenja i obučavanja. Međutim, mnogi ljudi ne vide jasno razliku između ove dvije aktivnosti i tretiraju ih kao sinonime. Jedina sličnost je što su ove dvije aktivnosti aspekti obrazovnog procesa [26].

Obuka je davanje informacija i znanja putem govora, pisane riječi ili drugih metoda demonstracije, na način koji upućuje korisnika. Obuka se više fokusira na razvoj novih vještina ili skupova vještina koje će se koristiti. Svi su imali iskustvo obuke kad ih je angažirala tvrtka. Prvo što svaki poslodavac mora učiniti je obučiti početnika kako bi mu dao potrebne upute o njegovim dužnostima, odgovornostima i testirao ga. Danas mnoge tvrtke traže zaposlenike koji se, iako nemaju potrebne vještine, mogu brzo obučiti. To je brže i štedi se više novca od traženja kandidata s odgovarajućim vještinama [26].

U osnovi, kroz obuku se ne želi preoblikovati ponašanje pojedinca, već poanta je naučiti zaposlenika kako se to radi kako bi oni mogli sami provesti postupak. U idealnom slučaju, okruženje za elektroničko učenje upotrebljavat će i principe učenja i osposobljavanja u svom nastavnom planu [23].

Učenje je proces apsorbiranja informacija kako bi se povećale vještine i sposobnosti i iskoristile u raznim kontekstima. Ovaj je postupak posebno orijentiran za postizanje određenog cilja, čime se usvajaju potrebne vještine i informacije. Međutim, učenjem se dolazi do znanja koje se može koristiti i sada i u budućnosti i koje se ne može oduzeti. Čak se može reći da osoba postaje dobro opremljena određenim vještinama i znanjem koje će joj pomoći da se nosi s izazovima svaki put kad se suoči s njima [26]. Generalno, učenje je proces i skup aktivnosti kako nekoga osposobiti da se bavi ne samo današnjim problemima, već ga pripremiti za kreativan pronalazak načina za rješavanje budućih problema [23].

Razvojem tehnologija obučavanje i učenje su dobili nova značenja. Pojavio se fenomen učenja na daljinu koji pomaže studentima da steknu kvalifikaciju i diplomu s bilo kojeg sveučilišta. Online obrazovanje u posljednje vrijeme dobiva mnogo pozornosti jer je vrlo pristupačno i ugodno za mnoge ljude [26].

3. 3. Prednosti i nedostaci elektroničkog učenja

Prednosti elektroničkog učenja su:

1. U slučaju učenja licem u lice, mjesto ograničava posjećenost grupe korisnika koji imaju mogućnost sudjelovanja u tom području, a u slučaju vremena ograničava one koji samo mogu pohađati u specifično vrijeme. S druge strane, elektroničko učenje olakšava učenje bez organiziranja kada i gdje te mogu biti prisutni svi koji su zainteresirani za tečaj [27].

2. Dizajniranje tečaja na način koji ga čini interaktivnim i zabavnim korištenjem multimedije povećava ne samo faktor angažiranosti, već i trajanje predmetnog gradiva. Postoji mogućnost komuniciranja s ljudima na forumima, dijeljenja svog napretka na društvenim mrežama i tako dalje [27].

3. Ovo je usmjereno i na korisnike i na učitelje, ali postoji velika vjerojatnost za plaćanje povećeg iznosa novca kako bi se nabavilo ažurirane verzije udžbenika za obrazovnu instituciju. Kod elektronskog učenja zamjena i nadogradnja digitalnih sadržaja je znatno jednostavnija u odnosu na tiskani materijal. Još jedan primjer je da obrazovne institucije ne moraju korisnicima pružiti prostor za učenje i neke dodatne materijale za njihovo obrazovanje. Isto tako korisnici ne moraju nigdje putovati kako bi stekli nova znanja [27].

Uzimajući u obzir ove prednosti i mnoge druge, postaje očito razlozi zašto trenutni trendovi elektroničkog učenja pokazuju znatan rast u industriji. Prem izvoru [28] očekuje se da će svjetski prihodi od elektroničkog učenja do 2025. narasti na 325 milijardi dolara, što je izvanredno s obzirom na činjenicu da je taj broj bio tri puta manji - 107 milijardi dolara u 2015. godini.

Nedostaci elektroničkog učenja su:

1. Dobar primjer nedostatka učenja preko Interneta je da je praktične vještine nešto teže pokupiti preko Interneta. Na primjer, iako je lako podijeliti informacije o izradi stola tako da se snimi video i objasni, praktično iskustvo je neophodno [27].

2. Elektroničko učenje zahtijeva upotrebu računala i ostalih uređaja što znači naprezanje očiju, loše držanje i drugi fizički problemi mogu utjecati na korisnika [27].

3. Za neke korisnike obrazovna institucija nije samo mjesto na kojem mogu učiti - to je i mjesto na kojem se dolaze družiti, sklapati nova prijateljstva i naučiti nešto više od svojih profesora. Učenje putem Interneta većinom je samostalan rad i zbog toga neki korisnici se mogu osjećati izolirano [29].

4. Zbog toga što se može učiti ugodnim tempom, za neke korisnike može biti problematično. Iako su neki dobri u samoorganizaciji, neki to ne mogu bez jasnog roka za pisanje seminarskih radova i potrebe da profesoru prijave o svom napretku [29].

3. 4. Budućnost elektroničkog učenja

Budućnost elektroničkog učenja ima potencijal rasta pošto brzine Internetske veze se povećavaju, a s tim se stvaraju i mogućnosti za više multimedijskih metoda obuke. Kako sve više obrazovnih ustanova, korporacija i online učenika širom svijeta počinje prepoznavati važnost mrežnog učenja, njegova će se uloga u obrazovanju tek nastaviti razvijati. Tehnologije poput socijalnih medija također stalno transformiraju obrazovanje. Iako je svijet obrazovanja preko Interneta neosporno dinamičan svijet, mnogi korisnici kojima je učenje preko Interneta neugodno i dalje preferiraju tradicionalne metode podučavanja na koje su navikli. Svi korisnici imaju jedinstvene stilove učenja i učenje preko Interneta vjerojatno nikada neće biti rješenje obrazovanja za sve. Uz sve istaknuto, nema sumnje da su ovo početci obrazovanja nove ere [28].

4. WEB DIZAJN

Dizajn je proces prikupljanja ideja i estetskog uređenja vođen određenim principima [30]. Obično se odnosi na aspekt korisničkog iskustva web stranica, a ne na razvoj softvera. Web dizajn nekada je bio fokusiran na dizajniranje web stranica za desktop preglednike, ali od sredine 2010-ih dizajn za mobilne preglednike postaje sve važniji [31].

4. 1. Elementi dizajna

Web dizajn također se preklapa s grafičkim dizajnom, što ih čini vrlo bliskim. Zbog toga bi svaki web dizajner trebao poznavati osnovne principe dizajniranja. To uključuje sljedeće principe [32]:

TEORIJA BOJA

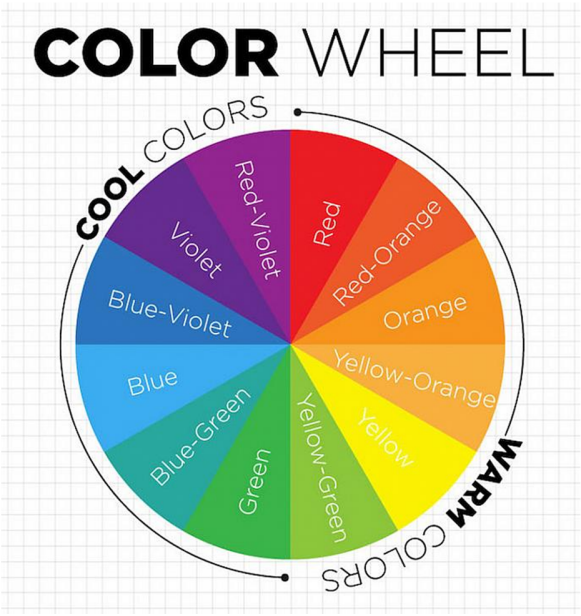
Ljudi odluče hoće li im se proizvod svidjeti u 90 sekundi ili ne, a 90% te odluke temelji se isključivo na boji [34]. Teorija boja je znanost i umjetnost korištenja boja. Objašnjava kako ljudi doživljavaju boju i vizualne efekte kada se boje međusobno miješaju, podudaraju ili međusobno razlikuju. Teorija boja također uključuje poruke koje boje komuniciraju. Boja je percepcija. Predmeti odražavaju svjetlost u različitim kombinacijama valnih duljina. Ljudski mozak prikuplja kombinacije valnih duljina i prevede ih u fenomen koji se naziva bojom [34].

Generalno, postoje dva osnovna načina miješanja boja. Suptraktivni model miješanja je kada tinte, boje ili pigmenti formiraju nove boje apsorbiranjem nekih dijelova vidljivog spektra. Aditivni model započinje tamnom i koristi različite boje svjetla pomiješane zajedno kako bi se postigla bijela [35].

Primarne boje su nijanse koje se mogu miješati da bi se dobila široka paleta boja. Većina ljudi poznaje crvenu, žutu i plavu (engl. Red, Yellow, Blue - RYB) kao primarne boje koje se ne mogu dobiti miješanjem drugih boja. Također RYB primarne boje su primjer modela

suptraktivnog miješanja. Još jedan primjer suptraktivnog miješanja boja je model svijetloplava, crvenoljubičasta i žuta (engl. Cyan, Magenta, Yellow - CMY). Posljednji korišten model je crvena, zelena i plava (engl. Red, Green, Blue - RGB) i koristi aditivni model miješanja boja [35].

Kotač u boji (engl. color wheel) sastoji se od tri primarne boje (crvena, žuta, plava), tri sekundarne boje (dobivene miješanjem primarnih boja, zelena, narančasta, ljubičasta) i šest tercijarnih boja (boje izrađene od primarnih i sekundarnih boja, poput plave-zelena ili crveno-ljubičasta). Kad se povuče vertikalna crta kroz sredinu kotača kao što je prikazano na slici ispod, razdvoje se tople boje (crvena, narančasta, žuta) od hladnih boja (plava, zelena, ljubičasta). Tople boje se uglavnom povezuju s energijom, svjetlinom i aktivnošću, dok se hladne boje često poistovjećuju s mirom, tišinom i spokojem. Pomoću kotača u boji dizajneri razvijaju shemu boja za marketinške materijale. Slijede tri najčešće sheme koje se koriste [34].



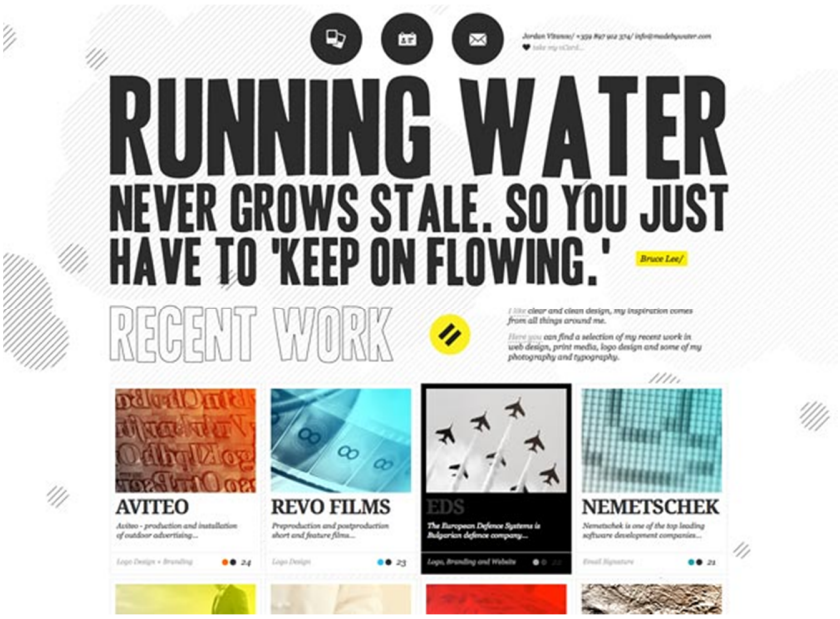
Slika 4: Kotač u boji [33]

Komplementarne boje su bilo koje dvije boje koje se nalaze jedna nasuprot drugoj, kao što su crvena i zelena te crveno-ljubičasta i žuto-zelena. Budući da postoji oštar kontrast dviju boja, one stvarno mogu istaknuti slike, ali prekomjerno korištenje može postati zamorno. Analogne boje su sve tri boje koje su jedna uz drugu na kotaču s bojama, poput žuto-zelene, žute

i žuto-narančaste. Pri stvaranju analogne sheme boja jedna boja će prevladati, jedna će podržavati, a druga naglasiti. Trijadne boje su ravnomjerno raspoređene oko kotača s bojama i obično su vrlo svijetle i dinamične. Korištenje trijadne sheme boja u marketingu istovremeno stvara vizualni kontrast i sklad, čime se svaki predmet ističe, a cjelokupna slika dolazi do boljeg izražaja. S ovim jednostavnim znanjem o bojama i shemama boja, puno je lakše i efikasnije donositi odluke o robnoj marki [34].

KONTRAST

Crno i bijelo zajedno čine beskonačne nijanse sive. Isto tako, brojni različiti vizualni elementi mogu biti postavljeni jedan pored drugog kako bi stvorili primjere kontrasta koji privlače pažnju [36]. Primarna svrha kontrasta boja je da usmjeri pažnju korisnika. Ako se dvije boje međusobno razlikuju (na primjer, crna i bijela), imaju visoki kontrast, dok ako su vrlo slične (crvena i narančasta), imaju slab kontrast [37].



Slika 6: Primjer korištenja kontrasta u web dizajnu [33]

Sljedeći način kako se kontrast može ostvariti je razlika između prednjeg i stražnjeg plana. Prednji plan sadrži stavke i elemente koji su najvažniji za korisnika i jasno utvrđuje

njihovo mjesto u vizualnoj hijerarhiji dok stražnji plan služi kao postolje na koji su postavljeni elementi prednjeg plana. Ovaj način se koristi za naglašavanje važnih informacija [36].

Sljedeći oblik kontrasta je upotreba veličine te je primjer prikazan na slici iznad. Općenito se kontrast koristi u nekom ili drugom obliku za usmjeravanje pažnje korisnika. Veličina to čini tako što ljudskom oku daje do znanja da ako je nešto veće, automatski znači da je i bitnije. Kontrast oblika znači razlikovati stvari po svojoj fizičkoj formi u odnosu na ostale na stranici. Na osnovnoj razini to se može koristiti u stvarima poput dodavanja zaobljenih uglova gumbima, ali postavljanjem oštrijih rubova i ravnim linijama se može privući veća pažnja. Kontrast je jedno od najvažnijih načela u dizajnu i gotovo nikad ne može biti previše, pod uvjetom da se pravilno koristi [37].

TIPOGRAFIJA

Prvi dojmovi su trajni dojmovi [38]. Bez obzira da li se navedeni zaključak prihvaća ili ne, tipografija pomaže stvoriti doživljaj za korisnike prije nego što su čak pročitali tekst ili odabrali poveznicu. Tipografija uspostavlja način komunikacije i osobnost web stranice. Izbor slova može odrediti kako ljudi reagiraju na web stranicu. Dobra web tipografija ne odnosi se samo na lijepi vizualni tretman, već i na brzinu [38]. Izbori koji se donese u vezi s tipografijom mogu imati znatne posljedice u vezi s tim da li ljudi daju priliku da pročitaju tu web stranicu [39].

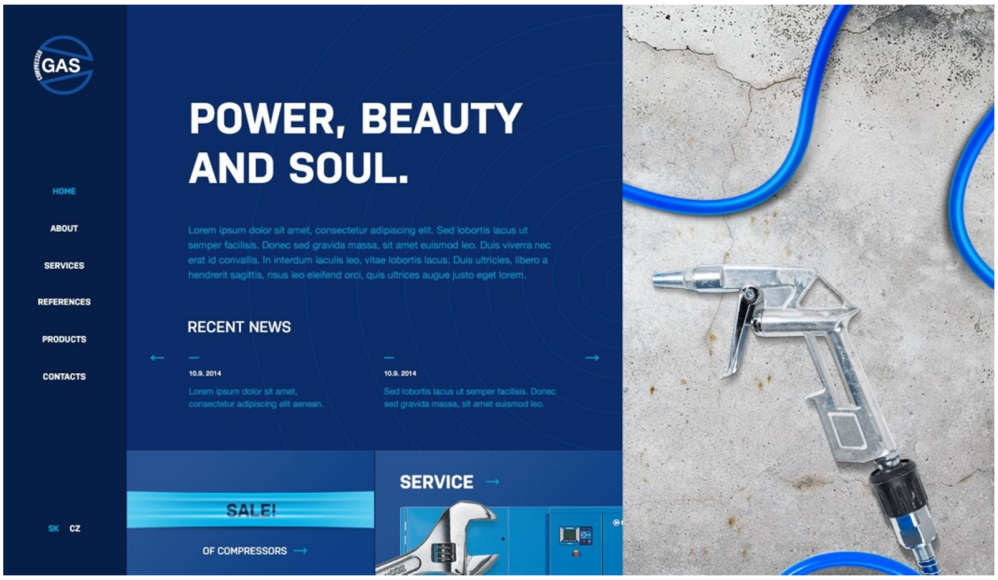
Psiholog Kevin Larson s MIT-a želio je vidjeti da li nešto tako naizgled banalno kao što je izbor i izgled fonta može emocionalno utjecati na korisnika. Među 20 volontera gdje su pola muškarci i pola žene, zamolilo ih se da ocijene verziju tjednog časopisa The New Yorker. Istraživači su otkrili da su volonteri koji su čitali lošije dizajniranu stranicu imali negativan utjecaj te su se osjećali loše. Ovo govori da formiranje pamtljivih asocijacija nije samo stvar uvjerljive priče nego ima i veze s izborom fonta [40].

Tipografija je još jedan način stvaranja određenog stila za web dizajn. Ako sve slike sadrže čiste linije, trebalo bi se upotrijebiti jednaki oblik fonta. Pomaže uspostaviti raspoloženje ili emociju. Na primjer, neozbiljniji i svjetliji oblik fonta signalizirao je korisnicima da je marka zabavna, mlada i ne shvaća sebe ozbiljno [39].

PONAVLJANJE ELEMENATA

Ljudska urođena sposobnost uočavanja promjena u oblicima, uzorcima i ponašanju oko nas također se podudara s našom sposobnošću da vidimo jednolikost u oblicima, bojama i uzorcima. To je naša sposobnost razumijevanja uzoraka, uz uvažavanje kontrasta, što nam omogućava brže učenje i brže prilagođavanje [42].

Elementi koji se mogu učinkovito ponoviti su boje, oblici, teksture, fontovi, grafika, slike, videozapisi, zaglavlja, podnožja, navigacija i slično. Ponovljeni elementi na dosljedan način pomažu unapređivanju organizacije web stranice i održati kontinuitet [43]. Primjer primjene ponavljanja je prikazan na slici ispod gdje se boja ponavlja, odnosno različite nijanse plave boje.



Slika 5: Primjer primjene ponavljanja u web dizajnu [41]

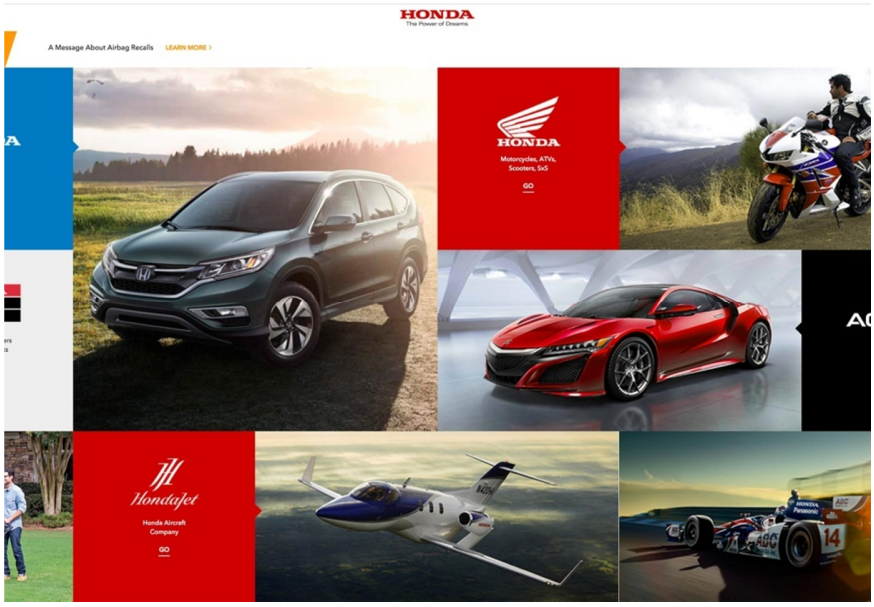
Dosljednost se dobiva iz ponavljanja sličnih elemenata, kao što su navigacijski odjeljci koji ostaju na istom mjestu bez obzira na kojem dijelu web stranice korisnik se nalazi. To pomaže posjetitelju da se osjeća ugodno što će svaka stranica reagirati na sličnu funkcionalnost i održava ravnotežu tijekom Internetskog iskustva. Ponavljana upotreba elemenata, koji imaju zajednički atribut, stvara ugodan vizualni uzorak, a male varijacije jednostavnog ponavljanja pojačat će posjetiteljevu znatiželju i pažnju [43].

RAVNOTEŽA ELEMENATA

Postizanje ravnoteže u web dizajnu zahtijeva malo razmišljanja, truda i vještina, ali to je apsolutno vrijedno toga [44]. Ravnoteža održava čitav dizajn usklađenim, a istovremeno postiže sve željene ciljeve. Važnost ravnoteže u procesu web dizajna ne može se podcijeniti, jer je jedino što sprečava web dizajnera da ne ispuni očekivanja korisnika. U slučaju da elementi web stranice nisu uravnoteženi s obzirom na vertikalnu os, rezultat neće biti ugodan [45].

Jedan od najčešćih primjera ravnoteže na koji se nailazi prilikom pregledavanja web stranica je simetrija. Simetrija je ugodna oku i stvara dizajn koji je estetski dobro organiziran i skladan. Simetrična ravnoteža definirana je postavljanjem elemenata jednako s obje strane vodoravne ili okomite središnje osi [44].

Poznata je činjenica da je teže oblikovati web stranice koje su asimetrično uravnotežene, zbog toga što elementi nisu uravnoteženi u skladu sa središnjim crtama dizajna. Uravnoteženje asimetričnog dizajna može se dobiti korištenjem različitih boja, tekstura ili veličina kao što je prikazano na slici ispod. Zanimljivija je od simetrije zato što je simetrija manjkava kada je u pitanju vizualna hijerarhija, dok su asimetrični dizajni povezani s tim [45].



Slika 8: Primjer primjene asimetrične ravnoteže [44]

Radijalna ravnoteža nastaje kada svi elementi dizajna na stranici izlaze u podjednakim točkama od središnje točke. Dakle, ako se stranica podijeli vertikalno ili vodoravno duž središnje osi, elementi obje strane bili bi podjednako udaljeni od središnje točke ili u blizini [44].

VIZUALNA HIJERARHIJA

Sadržaj u bilo kojem digitalnom izgledu stranica slijedi određenu hijerarhiju. Zaglavlja se pojavljuju iznad teksta. Izbornici se nalaze na vrhu, dnu, lijevo ili desno od zaslona. Dizajneri pokušavaju organizirati sadržaj tako da prvo predstave sadržaj s najvišim prioritetom, zatim isporučuju ostatak sadržaja s najnižim prioritetom. „Hijerarhija" je jednostavno prikladniji izraz za organiziranje od najbitnijeg prema najmanje bitnom [46].

Korisnici definiraju vizualnu hijerarhiju web stranice ili aplikacije. Stavka koja prva privlači pažnju nalazi se na vrhu hijerarhije. Na način na koji se opažaju informacije utječe nekoliko čimbenika koji pridonose načinu rangiranja hijerarhije. Slijede čimbenici poredani od najbitnijeg prema najmanje bitnom [46].

1. Veličina - što je element veći, to će više pozornosti privući u odnosu na manje elemente.
2. Boja - svijetle boje imaju veću vjerojatnost da privuku pažnju nego tamnije nijanse.
3. Kontrast - dramatično kontrastne boje privući će veću pažnju nego blago kontrastne boje.
4. Poravnanje - može stvoriti red između elemenata dizajna.
5. Ponavljanje - ponavljajući stilovi mogu gledatelju dati osjećaj da je sadržaj povezan [46].

Slika 9: Primjer Z i F uzorka [46]

Postoje uobičajeni uzorak za hijerarhiju koji se temelje na pokretima koje oči imaju tendenciju da naprave kada su predstavljene sa svježom stranicom.

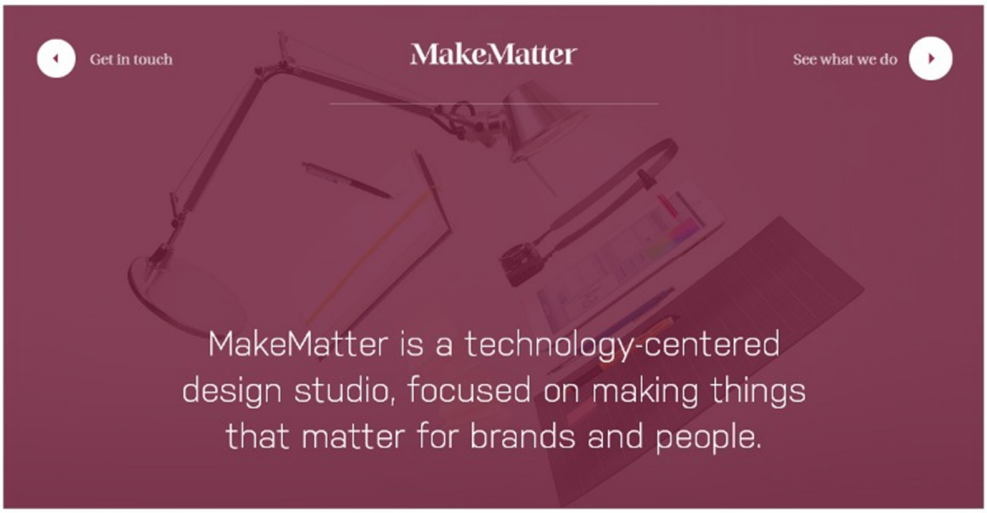
Na web stranicama s niskom razinom tekstualnog sadržaja uobičajen je Z uzorak smjer pregledavanja sadržaja. Korisnik vidi stranicu s tekstom i pregledava s gornje lijeve na gornju desnu stranu, a zatim kroz sadržaj (slijedom dijagonale) pogleda dolje lijevo, prije nego što se pomakne u donji desni kut stranice [46].

Dizajneri obično primjenjuju F uzorak na web stranicama koje uključuju tekstualni sadržaj i / ili video sadržaj. Korisnici započinju pregled sadržaja s lijeva na desno duž vrha, ali zatim pregledavaju sadržaj dolje s lijeve strane, tražeći vizualne tragove do informacija koje traže. Kad pronađu takav trag, pregledavaju sadržaj s lijeva na desno. Ponavljaju ovaj postupak sve dok ne dođu do kraja stranice [46].

NEGATIVAN PROSTOR

Postoje dvije vrste prostora u dizajnu, pozitivni i negativni prostor. Pozitivan prostor sadrži sve elemente web stranice. Negativni prostor (koji se često naziva i bijeli prostor) je sav prazan prostor oko sadržaja koji razdvaja elemente i pomaže u strukturi. Postoje dvije razine negativnog prostora: mikro i makro. Mikro negativni prostor povezan je s prostorom između malih elemenata poput linija, riječi i slova. Makro je negativni prostor između većih blokova ili elemenata. Obje su razine važne za ukupnu učinkovitost web dizajna. Slijede savjeti kako pravilo koristiti negativan prostor na web stranici [47].

Negativni prostor pruža korisnicima vrijeme da apsorbiraju podatke koje vide na stranici. Kada web stranica ima previše informacija i premalo prostora, posjetiteljima je izuzetno teško usredotočiti se na glavne informacije. Čitljivost sadržaja je vrlo važan aspekt u web dizajnu i negativni prostor može pomoći optimizirati. Pravilno iskorišten prostor između redaka, riječi i slova znatno povećava razumijevanje čitanja [48].



Slika 10: Primjer efikasnog korištenja negativnog prostora [47]

Negativni prostor nije samo prazan prostor između elemenata na stranici, već je ključni alat za izgradnju vizualne hijerarhije. Negativni prostor može sastaviti ili odvojiti elemente korisničkog sučelja tako da stvaraju učinkovit dizajn stranice [48]. Slika iznad prikazuje efektivan način korištenje negativnog prostora tako da je stavljen razmak između naslova,

kratkog opisa i oba gumba te jednakim razmakom s obje strane kratkog opisa. Ovako jednostavno postavljeni elementi omogućuju web stranici da „diše", odnosno omogućuje korisnicima da lakše apsorbiraju informacije .

4. 2. Razlozi važnosti web dizajna

Kad publika posjeti web stranicu, to im daje prvi dojam poslovanja. Korisnici će prosuditi o poslu u roku od nekoliko sekundi. U ovih prvih nekoliko sekundi želi se pozitivno utjecati na korisnike web stranica. Web dizajn je važan jer utječe na to kako korisnici percipiraju robnu marku. Dojam koji se ostavlja na njih može ih natjerati da ostanu na stranici i nauče o poslovanju ili napuste stranicu i okrenu se konkurentu. Dobar web dizajn pomaže da se zadrže potencijalni korisnici na svojoj stranici [49].

Ljudi mogu prosuditi kako će se prema njima postupati gledajući web stranicu [44]. Dizajn daje uvid u to kako se gleda na korisnike. Ako se ne uloži nikakav napor u dizajn web stranice, korisnici pretpostavljaju da se neće uložiti napor da im se pomogne. Važno je izgraditi povjerenje s korisnicima kako bi ostali na web stranici. Kada korisnici duže ostanu na web stranici, stvara se više mogućnosti za tvrtke da zabilježe potencijalne korisnike [49].

Pošto se želi izdvojiti od konkurencije, mora se uložiti vrijeme u web dizajn. Ako se posjeduje zastarjela web stranica, konkurenti će to iskoristiti. To znači da će se izgubiti potencijalne korisnike u odnosu na konkurenciju. Privući će više korisnika na svoje web stranice jer je njihova stranica privlačnija [49].

Web dizajn je važan jer pomaže u stvaranju dosljednosti na cijeloj stranici. Moraju se koristiti isti fontovi, stilovi i izgledi na svim stranicama na web stranici. Ako na svakoj stranici postoji drugačiji dizajn, zbog toga će web stranica izgledati neprofesionalno. Također čini kompliciranijem prepoznavanje robne marke jer korisnici neće znati koje boje treba povezati s robnom markom. Izgradnjom dosljednosti zadržava se više potencijalnih korisnike na web stranici i upoznaje se ih s poslovanjem [49].

4. 3. Potrebne vještine web dizajnera

Osim smisla za dizajn i poznavanja osnovnih elemenata dizajna, potrebno je još par vještina koje bi morao imati svaki web dizajner.

Poznavanje HTML-a (engl. Hypertext Markup Language) svodi se na kodiranje. Čini se da neki dizajneri misle da nije potrebno znanje o kodiranju, dok su drugi uporni da je znanje o kodiranju nužno kako bi se postalo cjelokupan web dizajner. U najmanju ruku, potpun dizajner trebao bi znati osnove HTML-a o kojem će kasnije biti više rečeno [32].

Korisničko iskustvo uobičajeno se skraćuje kao UX (engl. User Experience). Sve se „vrti" oko toga kako se korisnici osjećaju kada koriste web stranicu, te zbog toga je potrebno imati na umu perspektivu korisnika. Prilikom dizajniranja web stranice, mora se osigurati da će korisnici dobiti upravo ono što žele. Za postizanje tog cilja je potrebno obaviti istraživanje [50].

Kada se stvara web stranica, postoji mnogo stvari koje se moraju uzeti u obzir. Kada je riječ o korporacijskoj web stranici, vjerojatno su najvažniji poslovni zahtjevi. Razlog je taj što svaka korporativna web stranica ima određenu funkciju - dovođenje novih kupaca, povećanje izloženosti robne marke, više potencijalnih ponuda itd. Web dizajner mora razumjeti ove funkcije i omogućiti im da postignu željeni dizajn [51].

5. METODE IZRADE WEB STRANICE

U ovom poglavlju će biti objašnjena dva načina kako se može kreirati web stranica. Prvi je s CMS sustavom (engl. Content Management System) koji ne zahtijeva prethodno znanje. Drugi način je izrada web stranice od početka, što uključuje front-end i back-end. Ukratko rečeno, front-end se odnosi na vidljivi i/ili interaktivni dio stranice, dok back-end se odnosi na dio stranice koji nije vidljiv korisnicima te je odgovoran za spremanje,slanje i primanje podataka.

5. 1. Sustav za upravljanje sadržajem

Sustav za upravljanje sadržajem, odnosno CMS je softver koji pomaže korisnicima u stvaranju, upravljanju i izmjeni sadržaja na web stranici bez potrebe za specijaliziranim tehničkim znanjem [52]. Obično se koriste za upravljanje web sadržajem (engl. Web Content Management - WCM) i za upravljanje sadržajem poduzeća (engl. Enterprise Content Management - ECM) [54].

CMS je alat koji pomaže u izradi web stranice bez potrebe za pisanjem cijelog koda ili znanja o kodiranju. Umjesto formiranja vlastitog sustava za izradu web stranica, pohranu slika i drugih funkcija, sustav upravljanja sadržajem upravlja svim onim osnovnim infrastrukturnim funkcionalnostima kako bi se autor web stranice mogao usredotočiti na bitnije dijelove web stranice [52].

Postoje dva glavna dijela koja pomažu u stvaranju web stranice korištenjem CMS sustava. Aplikacija za upravljanje sadržajem (engl. Content Management Application - CMA) omogućuje trgovcima i drugim stvaraocima sadržaja da izravno rade sa sadržajem, bez potrebe za uključivanje IT odjela. Aplikacija za dostavu sadržaja (engl. Content Delivery Application - CDA) djeluje kao pomoćni dio web stranice, uzimajući sadržaj koji se unosi u predloške i pretvara ga u radnu web stranicu kojoj posjetitelji mogu pristupiti [53].

5. 1. 1. Tipovi sustava za upravljanje sadržajem

1. Open Source CMS

Pojam open source podrazumijeva da korisnici mogu besplatno koristiti, mijenjati ili dijeliti softver te je najčešće izvorni kod dostupan korisniku za prilagođavanje. Izuzetno popularni open source sustavi za upravljanje sadržajem koriste PHP (skriptni jezik na strani servera i alat za web razvoj). WordPress, Drupal i Joomla glavni su primjeri. Takve sustave za upravljanje sadržajem je moguće besplatno koristiti bez obzira na sadržaj web stranica. Također nije potrebno dopuštenje za modificiranje open source CMS sustava. Open source CMS su besplatni jer nema ugovora za potpisivanje, nema dugoročnih obveza, nema naknade za nadogradnje i nema naknade za licencu [54].

2. Vlasnički CMS

Mnoge organizacije prodaju licence za korištenje vlastitog CMS sustava (engl. Proprietary Content Management Systems). Vlasnička aplikacija znači da postoji vlasnik aplikacije, a korisnici sustava moraju kupiti licencu od vlasnika ili trebaju vlasničko dopuštenje za upotrebu. U većini slučajeva, čak i uz licencu, vlasnici licenci još uvijek ne mogu mijenjati CMS ili duplicirati aplikaciju, osim ako nisu kupili licencu za programere. Dva glavna nedostatka vlasničkog CMS sustava uključuju trošak licenci i budući da mnoge tvrtke koje pružaju usluge smještaja na web serveru (engl. web hosting) ne podržavaju vlasnički sustav upravljanja sadržajem, preostaju ograničene mogućnosti za postavljanje web stranice na Internet [54].

3. Softver kao usluga CMS

SaaS (engl. Software as a Service) CMS rješenja obično uključuju WCM softver, usluge smještaja na web serveru i tehničku podršku. Ta se virtualna rješenja temelje na modelu pretplate te se smještaju i održavaju u oblaku (engl. cloud). Cijene su obično određene na temelju broja stranica ili korisnika i uglavnom uključuju pohranu podataka i sadržaja, količinu prijenosa podataka i stalnu podršku [54].

5. 1. 2. Prednosti i nedostaci CMS sustava

Prednosti CMS sustava su :

1. Jedna od prednosti izbora CMS sustava je mogućnost brzog pokretanja novog projekta. U radu s CMS sustavom već su razvijene potrebne značajke i funkcionalnosti, što omogućava brži razvoj web stranica nego njihova ručna izrada [55].

2. Sadržaj stranica i programski kod su međusobno odvojeni. Korisnik koji želi napraviti izmjene sadržaja na web stranici ne treba znanje HTML-a, CSS-a (engl. Cascading Style Sheets) ili drugih jezika kodiranja, ali njihovo poznavanje daje više mogućnosti u radu [55].

3. Rad u CMS sustavu često uključuje nekoliko korisnika ili timova stoga znanja i vještine ovih korisnika možda nisu iste. Ograničavanje korisničkog pristupa u CMS sustavu vrlo je korisna značajka jer omogućuje kontrolu tko može pristupiti dijelovima web stranice i koje su radnje dopuštene [55].

Nedostaci CMS sustava su:

1. Najpopularniji CMS sustavi su prilagođeni što većem broju ljudi što znači da je uključena osnovna funkcionalnost koja je potrebna za stvaranje web stranice. U slučaju da postoje posebne potrebe ili zahtjevi za web stranicu, to možda neće biti moguće napraviti korištenjem CMS sustava [55].

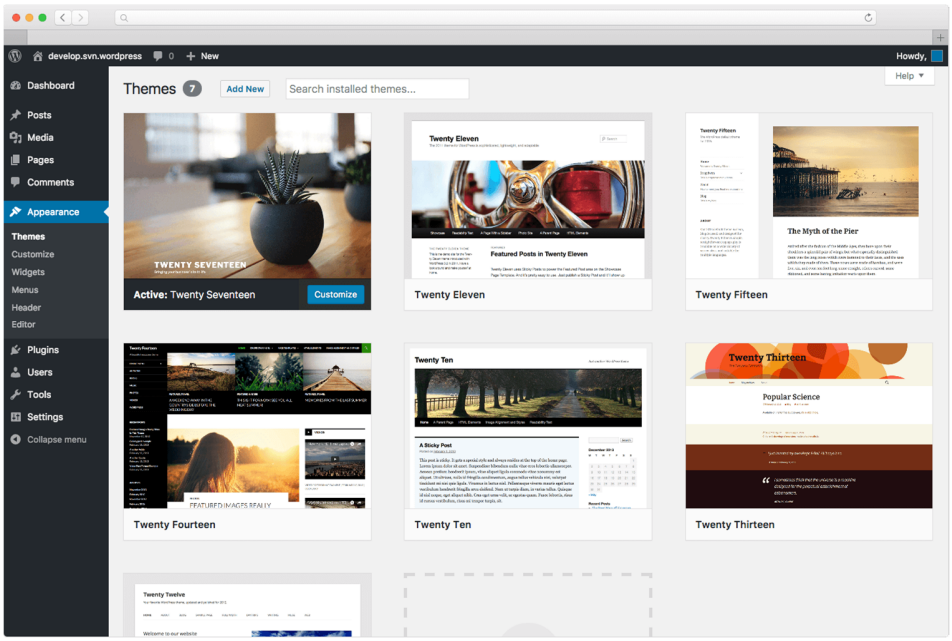
2. Korištenjem CMS sustava postoji veća mogućnost napada zlonamjernih korisnika. Najpopularnije CMS sustave koriste tisuće ljudi širom svijeta, pa ako zlonamjerni korisnici pronađu sigurnosni propust, to mogu iskoristiti protiv mnogih korisnika koji koriste sustav [55].

3. Većina CMS sustava dolazi s predlošcima dizajna kako bi se započelo s dizajnom sučelja web stranice. No iako se može prilagoditi većinu elemenata na web stranici, ponekad je možda potrebno napraviti kompromis. Razlog tome je način na koji je CMS izgrađen te se neki elementi ne mogu lako prilagoditi željenom dizajnu [55].

5. 1. 3. Primjeri CMS sustava

1. WordPress

WordPress je open source CMS napisan u programskom jeziku PHP te povezan s MySQL ili MariaDB bazom podataka [56]. Jednostavno i brzo se može izgraditi WordPress web stranica pomoću Gutenbergovog uređivača, a zatim ga prilagoditi bilo kojem od tisuća dodataka i tema dostupnih u službenom direktoriju WordPressa ili na drugim web stranicama. Započeo je kao jednostavan softver u svrhu bloga, ali evoluirao je kao potpuno funkcionalan CMS koristeći tisuće dodataka, widgeta i tema [54].

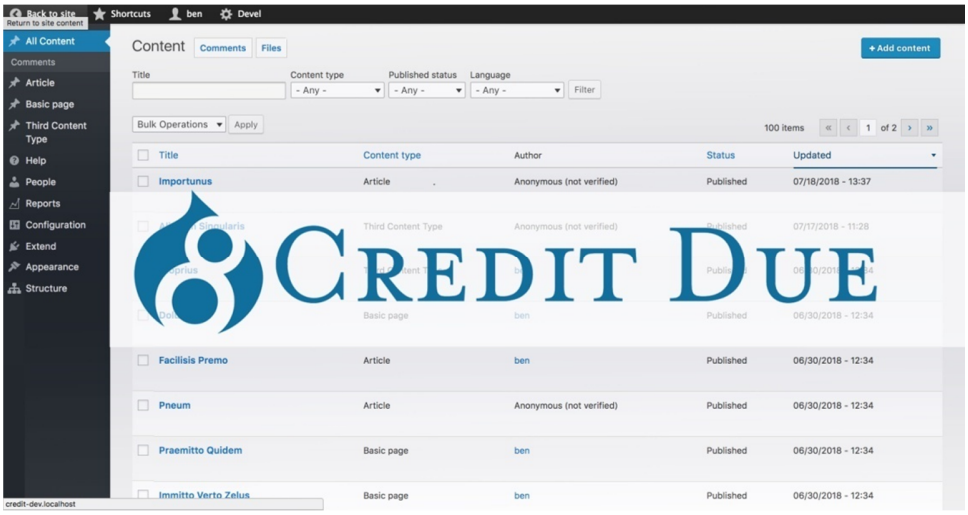


Slika 6: WordPress [57]

Zahvaljujući programu za brzu instalaciju i izvrsnoj dokumentaciji, izvrsna je platforma za početnike. Još jedna prednost ovog open source CMS sustava je ta što se najnovije verzije automatski ažuriraju s osnovnim programom i dodatcima iz pozadine, bez potrebe za preuzimanjem niti jedne datoteke [54].

2. Drupal

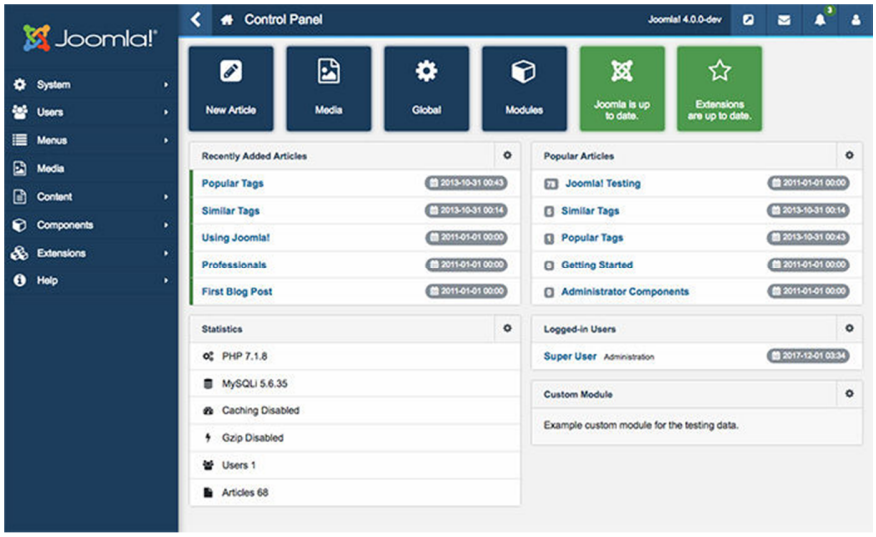
Drupal je open source CMS koji koriste milijuni ljudi i organizacija širom svijeta za izgradnju i održavanje web stranica. Drupal koristi Crveni križ, The Economist, BBC, NBC News, Cisco, Twitter i mnogi drugi. Objavljen je pod GNU Public License što znači da ima svojstvene prednosti - troškove, fleksibilnost, slobodu, sigurnost i odgovornost - koji nisu u skladu s vlasničkim softverom. Pruža korisničko sučelje koje omogućuje jednostavno stvaranje i objavljivanje sadržaja te dinamično dohvaćanje, filtriranje i prezentiranje sadržaja efikasnim alatima [58].



Slika 7: Jedna od mogućih tema za Drupal [59]

3. Joomla

Joomla je open source CMS koja povezuje web stranice s MySQLi, MySQL ili PostgreSQL bazom podataka kako bi se olakšalo voditelju web stranice upravljanje sadržajem. Joomla je besplatna platforma svima koji imaju želju za izgradnju dinamičkih web stranica bez obzira koja je svrha. Ima mogućnost obavljati zadatke u rasponu od korporativnih web stranica i blogova do društvenih mreža i e-trgovina. Korištena je od strane najprepoznatljivijih svjetskih marki, poput Harvarda, iHop i MTV [60].



Slika 8: Joomla [61]

5. 2. Ručna izrada web stranice od početka

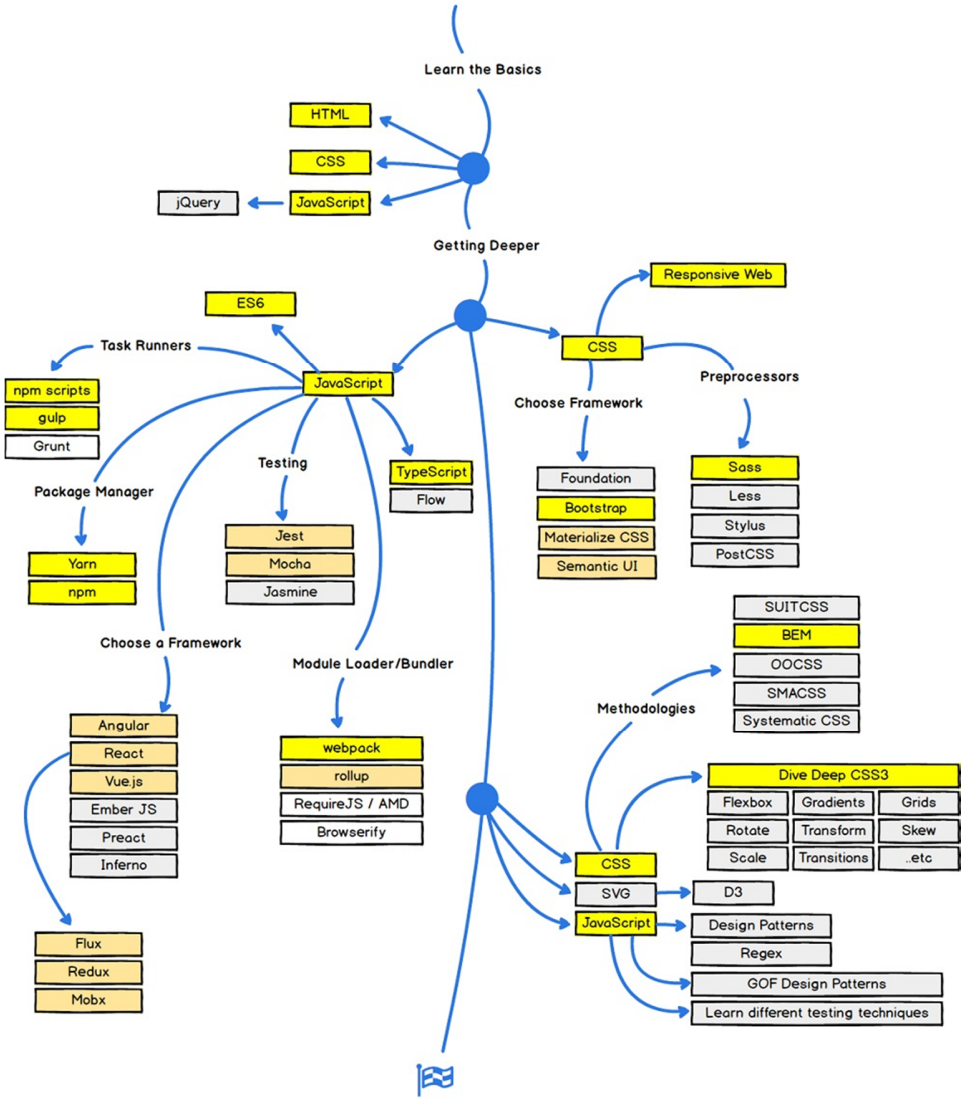
Izrada web stranice (engl. web development) je izgradnja i održavanje web stranica. Izrađena web stranica mora biti kvalitetna, raditi brzo i postići dobre rezultate uz besprijekorno korisničko iskustvo [62]. Posao web programera je uzeti web dizajn koji je kreirao klijent ili dizajnerski tim te transformirati ga u programski kod. Web programeri u osnovi moraju preuzeti jezik koji razumijemo i prevesti ga na jezik koji računalo razumije [63].

To zahtjeva puno vremena, truda i složeno razumijevanje različitih programskih jezika i načina na koji se koriste. Različite vrste programera specijalizirane su za različita područja, što znači da su veliki web projekti obično suradnja nekoliko različitih programera [63].

Kako bi se razumjelo što je web programer, ključno je definirati tri glavne vrste programera: front-end, back-end i full-stack. Front-end programeri odgovorni su za dijelove web stranice koje korisnici vide i u interakciji su s njima, back-end programeri odgovorni su za kod koji kontrolira kako se web mjesto učitava i pokreće, a full-stack programeri su kombinacija front-end i back-end programera [63].

5. 2. 1. Front-end programer

Front-end web stranice je dio s kojim korisnici komuniciraju. Sve što se vidi tijekom navigacije web stranicama, od fontova i boja do padajućih izbornika, kombinacija je HTML-a, CSS-a i JavaScripta kojim upravlja preglednik računala [64]. Izazov povezan s razvojem front-enda je taj što se alati i tehnike korišteni za stvaranje front-end web stranice stalno mijenjaju i tako programer mora stalno biti svjestan kako se polje razvija [65].



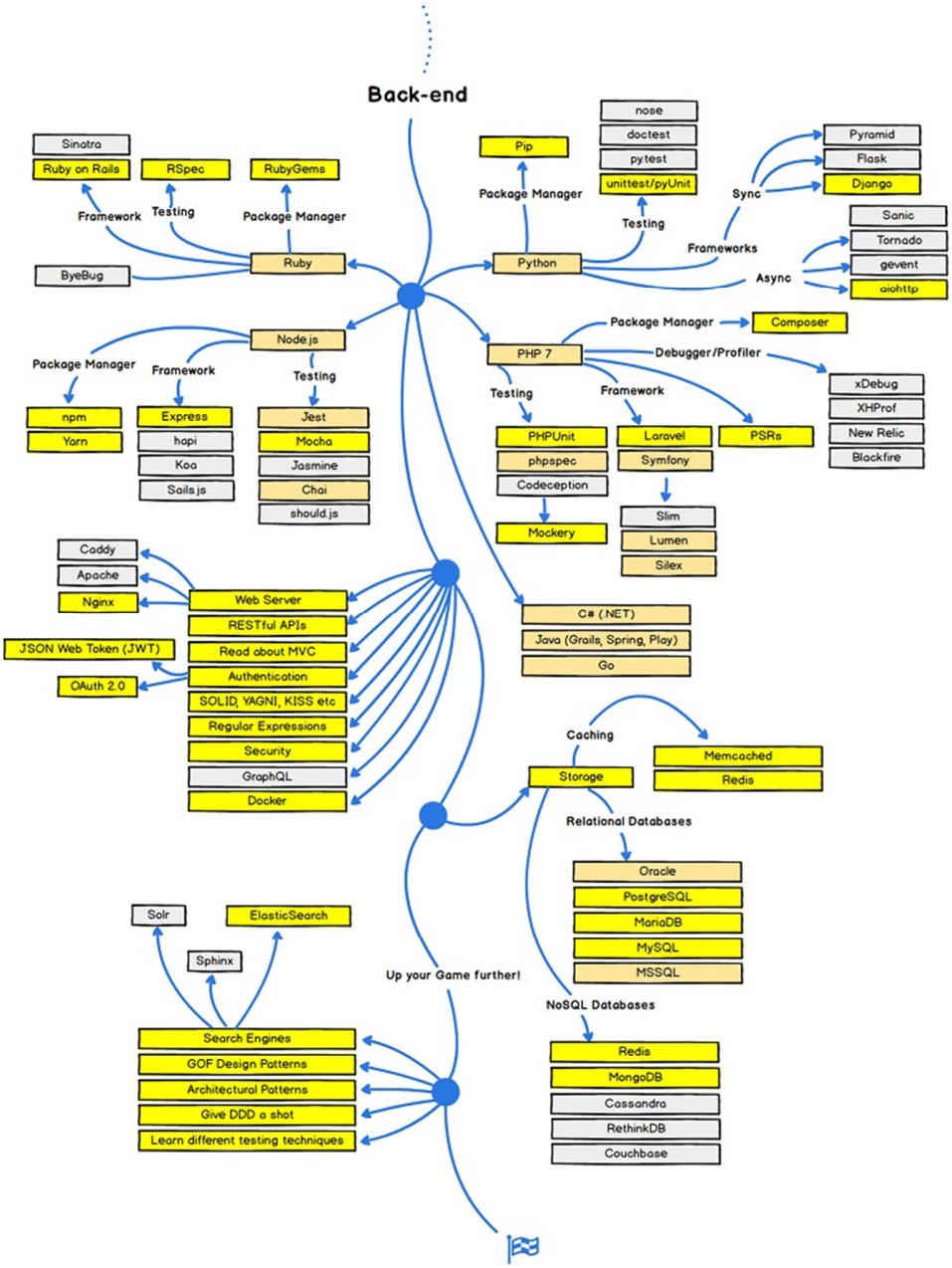
Slika 9: Putokaz za front-end programere [66]

To je dodatno kompliciranо činjenicom da korisnici sada koriste velik broj različitih uređaja s različitim rezolucijama zaslona, tako da web programer mora uzeti u obzir ove aspekte prilikom izgradnje web stranice. Moraju osigurati da se njihova web stranica ispravno pojavi u različitim preglednicima (engl. cross-browser), različitim operativnim sustavima (engl. cross-platform) i različitim uređajima (engl. cross-device), što zahtijeva pažljivo planiranje na strani programera [65].

Na slici iznad je prikazan putokaz za front-end programere te kreću s učenjem već tri spomenuta glavna jezika: HTML, CSS i JavaScript. Uz „tečno" korištenje ovih jezika, front-end programeri nastavljaju sa skupinom programskih biblioteka (engl. framework) poput Bootstrap, Foundation, Backbone, Angular JS i Ember JS, te knjižnice (engl. libraries) poput jQuery i LESS [64].

5. 2. 2. Back-end programer

Iako se može činiti da front-end programeri imaju težak posao osiguravajući da web stranica izgleda sjajno, dobro funkcionira i sadrži ispravan sadržaj, back-end programeri imaju kompleksniji zadatak. Dok su front-end programeri odgovorni za programiranje na strani klijenta, back-end programeri moraju se nositi sa serverom [63].



Slika 10: Putokaz za back-end programere [66]

Slično kao i za front-end programere, tako postoji i putokaz za back-end programere koji je prikazan na slici iznad. Kreće se s odabirom programskih jezika kao što su PHP, Ruby, Python i Java koji se koriste za izgradnju aplikacije, baze podataka poput MySQL, Oracle i SQL Server za pronalaženje, spremanje ili promjenu podataka. Sljedeći korak je znanje o skupovima

programskih biblioteka poput Zend, Symfony i CakePHP jer pomoću njih mogu doprinijeti web aplikacijama s čistim i prijenosnim kodom [64].

Prije nego što napišu kod, trebaju surađivati s poslovnim dijelom sveukupnog tima kako bi shvatili njihove posebne potrebe, zatim ih prevesti u tehničke zahtjeve i smisliti najučinkovitije rješenje za arhitekturu tehnologije [62]. Budući da različite web stranice imaju različite potrebe, back-end programer mora biti fleksibilan, sposoban stvoriti različite programe i mora imati temeljno razumijevanje jezika koje koriste [63].

5. 2. 3. Full-stack programer

Ako se traži brzi, jednostavni odgovor na pitanje „Što je web programer?", full-stack programer vjerojatno je najbliža stvar koja se može dobiti „kao odgovor". Full-stack programeri razumiju front-end i back-end strategije i procese, što znači da su savršeno pozicionirani za nadzor nad cijelim procesom [63]. Bez obzira na konkretne alate, ovisno o projektu ili klijentu koji je pri ruci, full-stack programeri trebaju biti dobro upoznati na svim razinama kako web funkcionira: postavljanje i konfiguriranje web servera, pisanje API-ja na strani servera, pisanje u JavaScript na strani klijenta [64].

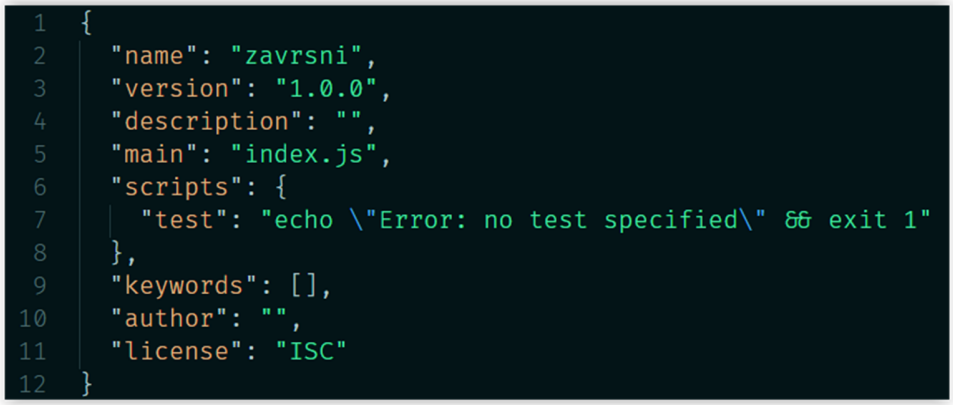
U slučaju (tehnički) jednostavnih web stranica koje nemaju veliki razvojni proračun, često se koristi full-stack programer za izradu čitave web stranice. U ovom je slučaju za njih iznimno važno imati cjelovito, dubinsko razumijevanje front-enda i back-enda i načina na koji rade [62]. Full-stack programeri često su zaposleni za nadgledanje velikih projekata za velike tvrtke za razvoj web stranica. Takve se pozicije vjerojatno isplaćuju više od standardnih web pozicija, što ih čini atraktivnijim za programere [63].

6. PRIMJER IZRADE WEB STRANICE

U ovom poglavlju će biti kratko objašnjena procedura pri izradi front-end djela web stranice pošto je to praktični dio završnog rada. Tema web stranice je robotika i cilj ove stranice bilo bi prikazati dio tema na jednom mjestu tako da se dobije jasnija slika o robotici.

Za početak je potrebno instalirati Node.js. Node.js je izvršno okruženje (engl. runtime environment) za programski jezik JavaScript, odnosno uključuje sve što je potrebno za izvršavanje programa napisanog u JavaScriptu. Prije su se JavaScript programi mogli pokrenuti samo pomoću preglednika, ali uz pomoć Node.js je moguće pokrenuti samostalni program na računalu [67]. U ovom primjeru glavni razlog za instalaciju Node.js je kako bi se mogao koristiti NPM (engl. Node Package Manager). NPM je odabran iz razloga zato što se smatra najboljim upraviteljem paketa za JavaScript, ima otvoren i dostupan kod te ima brojnu zajednicu korisnika što znači da je vrlo jednostavno naći rješenje problema [68].

Generalno upravitelj paketa je alat koji služi za automatizaciju procesa zadužene za rad s paketima. NPM dolazi sa Node.js instalacijom i u primjeru se najviše koristio za mijenjanje package.json datoteke i pokretanje raznih naredbi s klijentske strane [69]. JSON (engl. JavaScript Object Notation) je čitljiv format koji se koristi za pohranu i strukturiranje podataka te se sastoji od key/value parova [70]. Slika ispod prikazuje package.json datoteku te primjer key/value para bi na primjer bio: „name" kao key i „zavrsni" kao value. Kako bi se pristupilo package.json, u slučaju da se koristi MS Windows operativni sustav i rad u komandnoj liniji (CLI (engl. Command-Line Interface)), u komandnoj liniji se upiše npm init i tako se kreira datoteka na slici ispod.



Slika 11: package.json datoteka na početku projekta

Package.json je datoteka koja sadrži različite metapodatke vezana za projekt. Koristi se u svrhu davanja korisnih informacija NPM-u što omogućuje prepoznavanje projekta i rukovanje različitim zavisnostima (engl. dependencies) [71]. Zavisnost je kod od nekog drugog o kojem zavisi aplikacija [72].

Polje name je naziv projekta, a polje s author je ime autora projekta. Polje version koristi NPM kako bi osigurao instalaciju odgovarajuće verzije paketa. Polja s description i keywords pomažu drugim ljudima da razumiju o čemu se radi u tom projektu. Polje devDependencies su zavisnosti koje nisu potrebne za normalan rad, ali su preporučene ako se želi izmijeniti projekt. Polje dependencies je popis svih zavisnosti tog projekta koje su dostupne preko NPM-a i potrebne su za normalan rad [71].

Slika 12: package.json pri završetku projekta

Polje scripts se koristi za automatizaciju zadataka koji se ponavljaju. Prema slici gore, start se koristi za pokretanje lokalnog servera te se koristi pri izradi web stranica za automatsko ponovno učitavanje stranice na pregledniku. Polje build se koristi na kraju kada se završi s izradom da se smanji veličina podataka kako bi se stranica brže učitavala.

6. 1. Sustav za upravljenje modulima

Pisanje cijelog koda u samo jednoj datoteci nije efikasno iz razloga zato što je tako teže naći sitne greške i čini samo programiranje kompliciranijim. Stoga se u programskim jezicima, u ovom slučaju JavaScript, kod dijeli u manje dijelove, odnosno module [73]. Sustav za upravljanje modulima (engl. module bundler) služi za spajanje modula u jednu datoteku, prilikom čega vodi računa o zavisnostima. Za tu funkciju u ovom primjeru je korišten Webpack [69].

```
let webpack = require('webpack');
let HtmlWebpackPlugin = require('html-webpack-plugin');
let CopyPlugin = require('copy-webpack-plugin');

module.exports = {
    target: "web",
    devtool: "none",
    entry: {
        "theory": "./src/js/Learn/indexL.js",
        "login": "./src/js/Login/indexLogin.js",
        "template": "./src/js/Home/indexH.js"
    },
    plugins: [
        new HtmlWebpackPlugin({
            filename: "index.html",
            template: "./src/html/index.html",
            excludeChunks: ["theory", "login"],
            chunks: "template"
        }),
        new HtmlWebpackPlugin({
            filename: "theory.html",
            template: "./src/html/theory.html",
            excludeChunks: ["template", "login"],
            chunks: "theory"
        }),
        new HtmlWebpackPlugin({
            filename: "loginSign.html",
            template: "./src/html/loginSign.html",
            excludeChunks: ["template", "theory"],
            chunks: "login"
        }),
        new webpack.ProvidePlugin({
            $: 'jquery',
            jQuery: 'jquery'
        }),
        new CopyPlugin([
            { from: './src/img', to: 'imgs'}
        ])
    ],
```

Slika 13: Dio koda zaduženog za Webpack

Slika iznad prikazuje dio koda u kojem se koriste neke od mogućnosti Webpacka. Webpack je sustav za upravljanje modulima koji koristi konfiguracije pomoću kojih se opisuje kako će učitati datoteke s različitim ekstenzijama. Kada se pokrene korištenjem skripti iz package.json datoteke, sustav za upravljanje modulima čita ulaznu datoteku te skenira kod kako bi točno utvrdio što treba i o čemu ovisi svaki dio koda [74].

6. 2. Formiranje web stranice

HTML omogućuje stvaranje i strukturiranje odjeljaka, odlomaka, naslova, linkova i blokova za web stranice i aplikacije. Bez HTML-a, preglednik ne bi znao prikazati tekst kao elemente ili učitati slike. HTML nije programski jezik, što znači da nema sposobnost stvaranja

dinamičke funkcionalnosti te umjesto toga, omogućuje organiziranje i formatiranje dokumenata [75].

Većina HTML oznaka (engl. tags) ima početnu oznaku koja sadrži naziv oznake, atribute oznaka, završnu oznaku koja sadrži kosu crtu i ime oznake je zatvoreno. Za oznake koje nemaju završnu oznaku poput <img>, najbolja je praksa da se oznaka završi kosom crtom naprijed [75].

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Robots</title>
    </head>

    <body>
        <div class="header">
            <nav class="row">
                <ul class="login theory-login">
                    <li><a  href="./loginSign.html">Login / Sign up</a></li>
                </ul>
                <ul class="nav theory-nav">
                    <li><a  href="./index.html">Home</a></li>
                </ul>
            </nav>
        </div>

        <div class="container-theory">
            <div class="theory">
                <h2 class="theory-headline">Pick a topic you would like to learn more about</h2>
                <div class="theory-topics">
                    <div class="col span-1-of-4">
                        <a href="#" data-major="0">Electronic engineering</a>
                    </div>
                    <div class="col span-1-of-4">
                        <a href="#" data-major="1">Mechanical engineering</a>
                    </div>
                    <div class="col span-1-of-4">
                        <a href="#" data-major="2">Information engineering</a>
                    </div>
                    <div class="col span-1-of-4">
                        <a href="#" data-major="3">Computer <br>science</a>
                    </div>
                </div>
            </div>

            <div class="theory-topics-data"> </div>
            <div class="theory-topics-section"> </div>
        </div>

        <script type="text/javascript" src="theory.cbab746d7898ca9f461e.js"></script>
    </body>
</html>
```

Slika 14: HTML kod za jednu od tri stranice

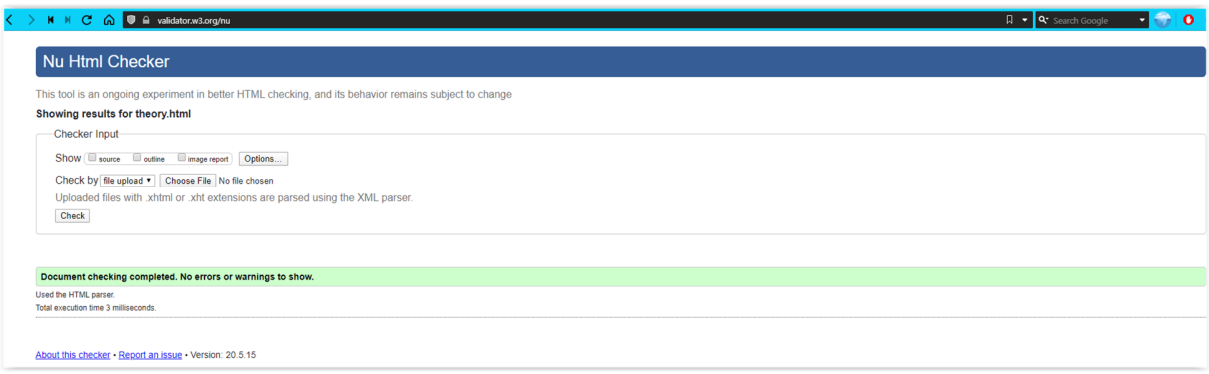
Početkom 90-tih kad se HTML tek počeo koristiti tzv.doctype je trebao djelovati kao poveznica na skup pravila koja HTML stranica mora slijediti da bi se smatrala dobrim HTML kodom. Međutim, ovih dana su izgubili na važnosti te ih treba uključiti kako bi sve ispravno funkcioniralo [76].

Element <html> „zatvara" sadržaj unutar tagova <html> i </html>, te se naziva korijenski element. Element <head> djeluje kao spremnik za naslov dokumenta, skup znakova, stilove, skripte i druge metapodatke te se ne prikazuju posjetiteljima stranice. Element <body> sadrži sav sadržaj koji se želi prikazati korisnicima kada posjete web stranicu, bilo da je riječ o tekstu, slikama, videozapisima ili nečemu drugom [76].

Dvije su važne kategorije elemenata u HTML-u: block-level i inline elementi. Block-level elementi se pojavljuju u novom redu i najčešće su strukturni elementi na stranici, kao na primjer odlomci, izbornici itd. Inline elementi se nalaze na razini bloka te okružuju samo male dijelove sadržaja. Obično se pojavljuju unutar odlomka kao na primjer <a> element [76].

Element se koristi za grupiranje sadržaja tako da se može lako stilizirati koristeći atribute poput class ili id [77]. Element <ul> služi za grupiranje predmeta koje nemaju neki poredak [78], a <li> element za numerirane stavke [79]. Oznake <h1> do <h6> koriste se za definiranje HTML naslova, a <a> određuje poveznicu koji se koristi za povezivanje s jedne stranice na drugu.

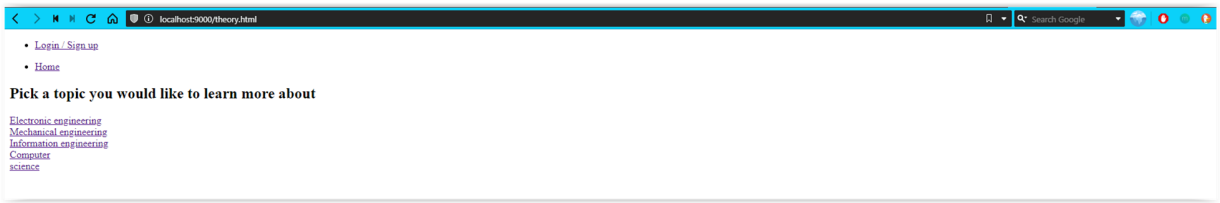
Atributi data- * koristi se za pohranu privatnih podataka na stranicu, što će kasnije biti više objašnjeno. Atribut href određuje poveznicu koja ima putanju prema web stranici [76]. Element <script> sadrži skriptne naredbe ili upućuje na vanjsku datoteku kroz atribut src.



Silka 15: Primjer stranice za provjeru HTML koda

Na slici iznad je prikazano korištenje web stranice u svrhu provjere prijašnjeg HTML koda. Kod pisanja HTML koda i izrade web stranica, HTML validator može se koristiti za provjeru je li HTML kod ispravan i da li slijedi W3C standarde postavljene za skriptiranje

HTML-a. HTML validator može provjeriti sve HTML oznake i pronaći bilo koji nevažeći kod ili bilo koji kod koji treba ažurirati kako bi bolje ispunio W3C standarde [80].

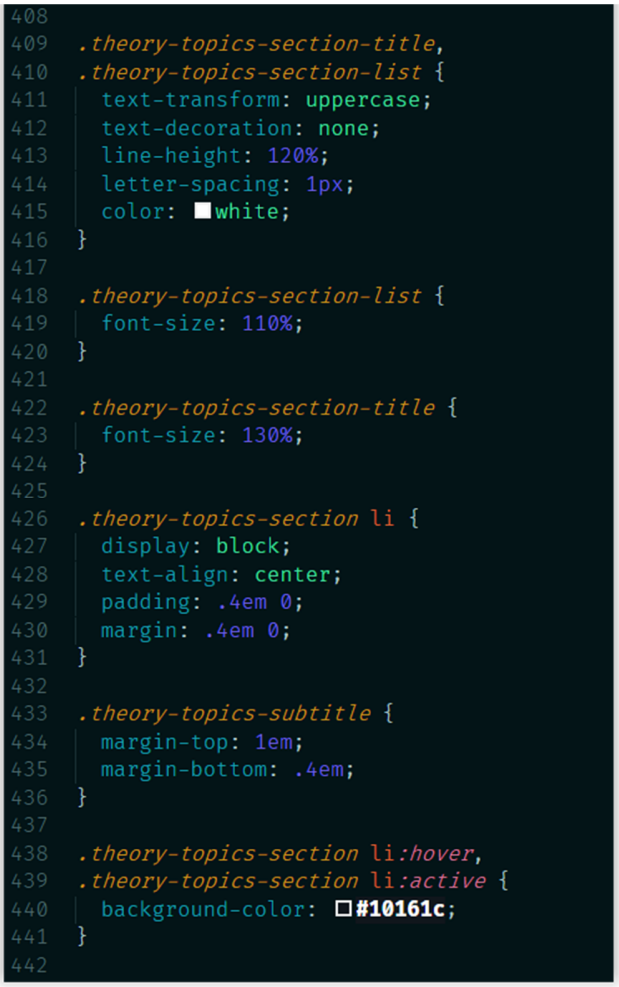


Slika 16: Osnovni stilovi preglednika

Na slici iznad su prikazani postavljeni stilovi preglednika. Preglednik stavlja ove osnovne stilove kako bi bili sigurni da će biti čitljivi, čak ako se i ne navedu određeni stilovi. Međutim, sve web stranice bi izgledale prilično slično te se korištenjem CSS-a može točno odrediti izgled HTML elemenata u pregledniku i tako postići željeni dizajn [81].

6. 3. Formatiranje sadržaja web stranice

CSS je stilski jezik koji se koristi za opisivanje dokumenta pisanog u HTML-u ili XML-u (engl. Extensible Markup Language). CSS se može koristiti za osnovno oblikovanje teksta dokumenta, na primjer za promjenu boje i veličine naslova. Može se koristiti za stvaranje izgleda, na primjer pretvaranje jednog stupca teksta u izgled s glavnim sadržajem i bočnom trakom za srodne informacije. Može se koristiti i za efekte poput animacije [81].



```
408
409  .theory-topics-section-title,
410  .theory-topics-section-list {
411    text-transform: uppercase;
412    text-decoration: none;
413    line-height: 120%;
414    letter-spacing: 1px;
415    color: white;
416  }
417
418  .theory-topics-section-list {
419    font-size: 110%;
420  }
421
422  .theory-topics-section-title {
423    font-size: 130%;
424  }
425
426  .theory-topics-section li {
427    display: block;
428    text-align: center;
429    padding: .4em 0;
430    margin: .4em 0;
431  }
432
433  .theory-topics-subtitle {
434    margin-top: 1em;
435    margin-bottom: .4em;
436  }
437
438  .theory-topics-section li:hover,
439  .theory-topics-section li:active {
440    background-color: #10161c;
441  }
442
```

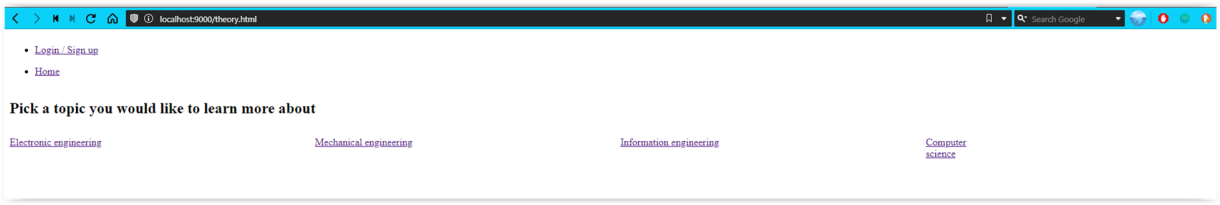
Slika 17: Dio CSS koda

Na slici iznad je prikazan dio koda i primjer kako se primjenjuju pravila u CSS-u. Pravilo započinje sa selektorom koji opisuje kojim će elementima u dokumentu pravilo odgovarati. Pripadajući elementi imat će definirani stil pravila primijenjen na njih [81].

U primjeru na slici iznad se koristio selektor koji označuje klasu u HTML dokumentu. On se koristi tako da se ispred naziva klase stavi točka, na primjer .theory-topics-section. U slučaju da to nije kraj, kao na primjer .theory-topics-section li, znači da pripadajući stilovi odgovaraju elementima li s klasom .theory-topics-section [82].

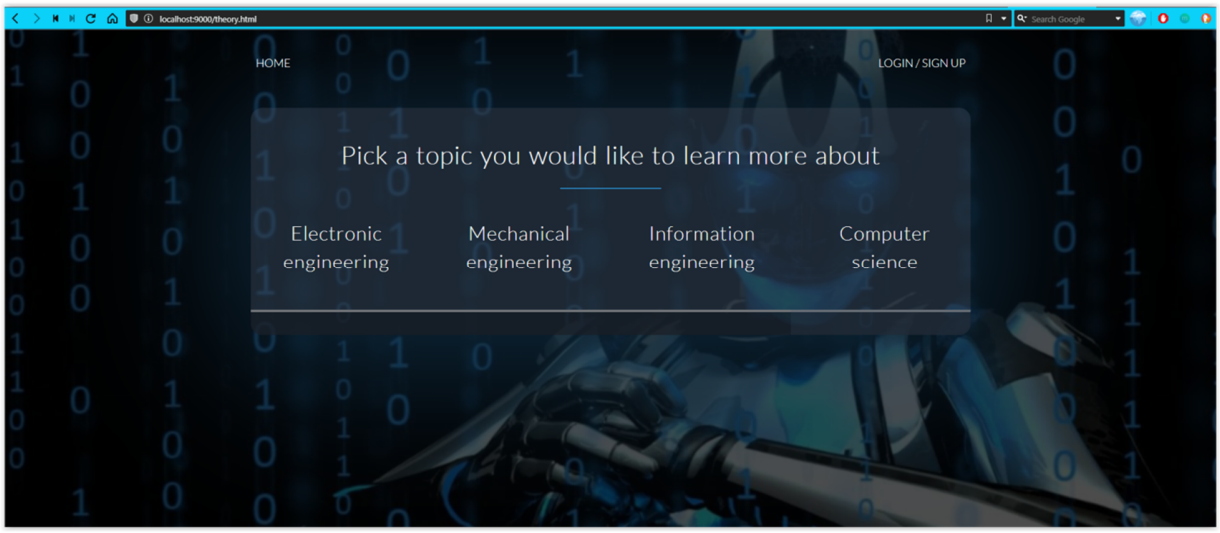
CSS pseudo klasa je riječ dodana selektoru koji određuje posebno stanje odabranih elemenata. U ovom primjeru to je :hover i :active. :hover se može koristiti za promjenu kada

pokazivač miša pređe preko tog elementa. :active predstavlja element koji je korisnik aktivirao [83]. Zatim slijede vitičaste zagrade, unutar kojih se nalazi jedan ili više stilova koji su u parovima: svojstvo i vrijednost, kao na primjer font-size: 110%. CSS svojstva imaju različite dopuštene vrijednosti, ovisno o tome koje je svojstvo korišteno.



Slika 18: Primjena Grid poretka

Kako bi se elementi postavili jedan do drugog koristio se dodatak Responsive Grid System koji je prikazan na slici iznad. Responsive Grid System je osnovni prilagodljiv (engl. responsive) sustav koji omogućuje postavljanje elemenata u stupce. Generalno prilagodljiv web dizajn znači da se prilagođava raspored web stranice za sve moguće rezolucije uređaja.



Slika 19: Kombinacija HTML-a i CSS-a

Slika iznad prikazuje rezultat korištenja HTML-a i CSS-a. Sljedeće je dodavanje interaktivnih elemenata na ovu web aplikaciju pomoću programskog jezika JavaScript.

6. 4. Programski jezik JavaScript

JavaScript (ukratko „JS") je dinamički programski jezik čijom primjenom se nad HTML dokumentu postiže interaktivnost [84]. U većini slučajeva JavaScript se koristi za izradu interaktivnih elemenata za web stranice, poboljšavajući korisničko iskustvo. Elementi poput izbornika, animacija, kontrole za upravljanje video sadržajem, interaktivnih karata i jednostavnije igara u pregledniku mogu se s JavaScriptom stvoriti brzo i jednostavno [85].

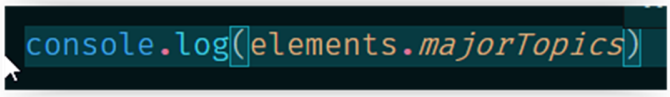
Java i JavaScript su različiti programski jezici i jedina sličnost im je riječ „Java". Različito su napisani i izvedeni, te imaju drukčiju namjenu. Što se tiče razvoja web stranica, Java se koristi za back-end aplikacija, dok se JavaScript koristi za front-end.

Na sljedećim stranicama slijedi kratak opis aplikacije koja je kreirana kao praktični dio završnog rada.

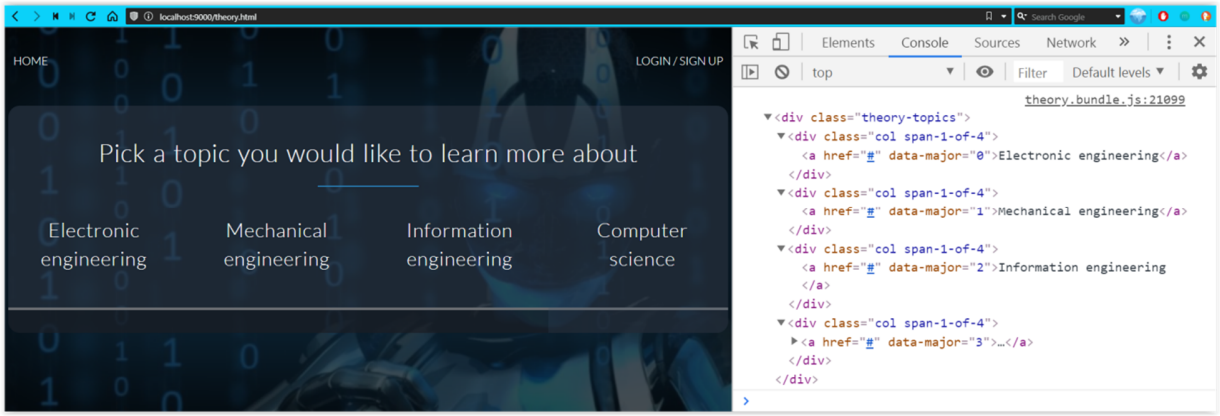
```
import uniqid from 'uniqid';

export const elements = {
    majorTopics: document.querySelector('.theory-topics'),
    topicsList: document.querySelector('.theory-topics-section'),
    lecture: document.querySelector('.theory-topics-data')
}

export const state = {};

export const addIdToSection = data => {
    data.forEach(cur => {
        cur.sections.forEach(cur => {
            cur.forEach(cur => {
                cur.id = uniqid();
            })
        })
    })
    return data;
}
```

Slika 20: Funkcija za dodavanje jedinstvenih brojeva

Na slici iznad je prikazan jedan od modula koji se koristi za dodavanje jedinstvenih brojeva podatcima. QuerySelector metoda vraća prvi element koji odgovara specificiranim CSS odabirom, u ovom slučaju su to klase. Console.log metoda prikazuje poruku u konzoli te je korisna u svrhu testiranja. Primjer je prikazan na slikama ispod.

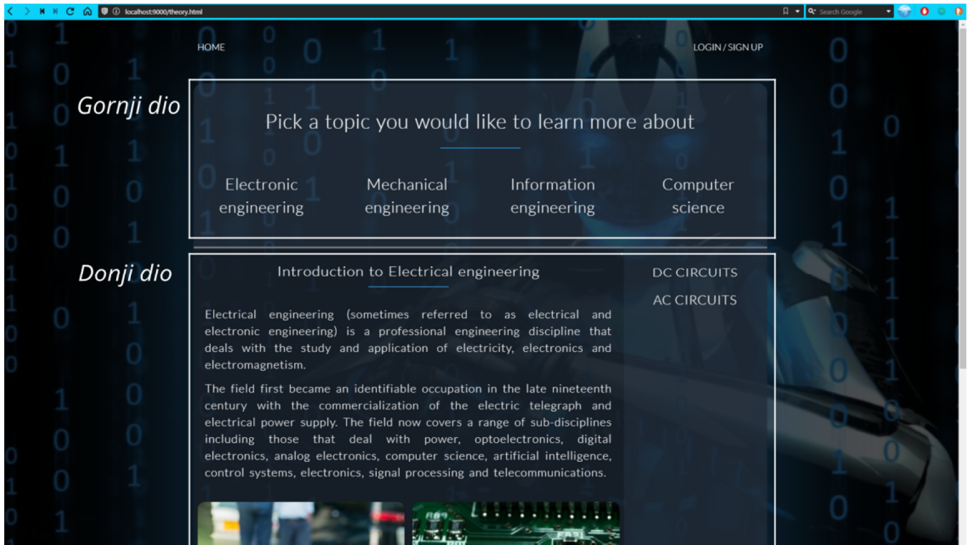


Slika 21: console.log metoda

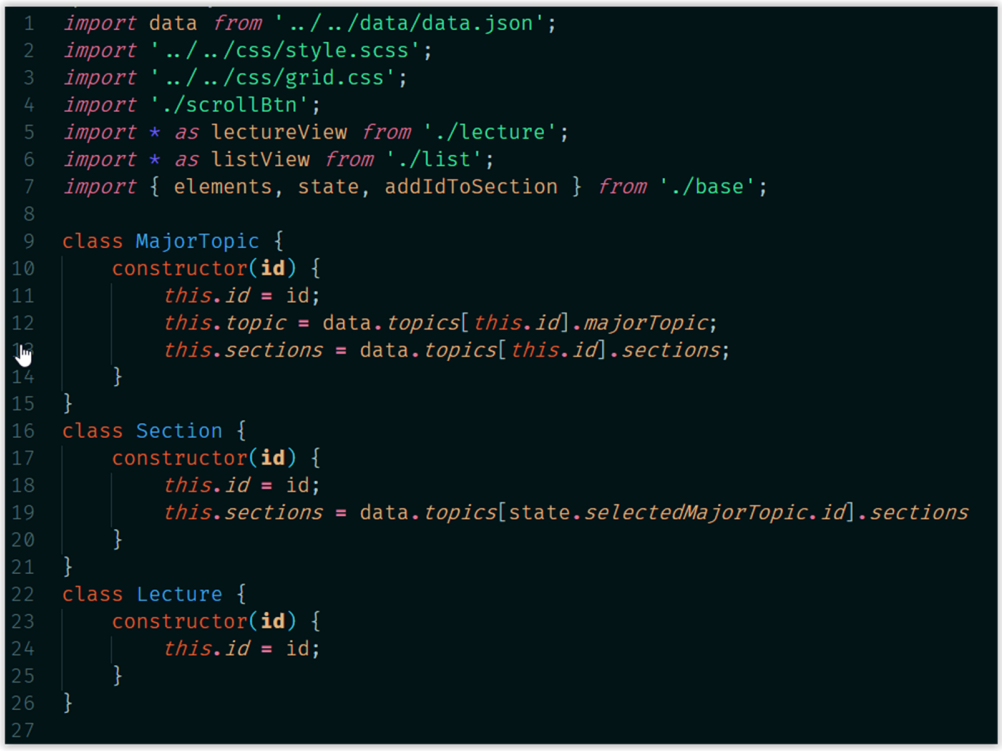


Slika 22: Rezultat QuerySelector metode prikazane na konzoli

Dobivene vrijednosti od querySelector metode se pohranjuju u objekt. Radi se na prezentirani način ako dođe do promjene naziva klase tada se promjena bilježi na jednom mjestu. Konstanta state je stanje aplikacije u bilo kojem trenutku. Funkcija addIdToSection služi za dodavanje jedinstvenog broja podacima koji služe za identifikaciju podataka.



Slika 23: Raspored aplikacije



```
import data from '../../data/data.json';
import '../../css/style.scss';
import '../../css/grid.css';
import './scrollBtn';
import * as lectureView from './lecture';
import * as listView from './list';
import { elements, state, addIdToSection } from './base';

class MajorTopic {
    constructor(id) {
        this.id = id;
        this.topic = data.topics[this.id].majorTopic;
        this.sections = data.topics[this.id].sections;
    }
}
class Section {
    constructor(id) {
        this.id = id;
        this.sections = data.topics[state.selectedMajorTopic.id].sections
    }
}
class Lecture {
    constructor(id) {
        this.id = id;
    }
}

```

Slika 24: Početak glavnog koda

Izraz import se koristi za uvoz drugih modula. Izraz class je specijalna vrsta funkcije koja se ne može pozvati (engl. invoke), nego se koristi za stvaranje objekta pomoću izraza new. Svrha stvaranja objekta preko identifikacijskog broja je dolazak do drugih korisnih vrijednosti i spremanje u konstantu state tako da se mogu podaci kasnije koristiti bilo gdje u kodu.

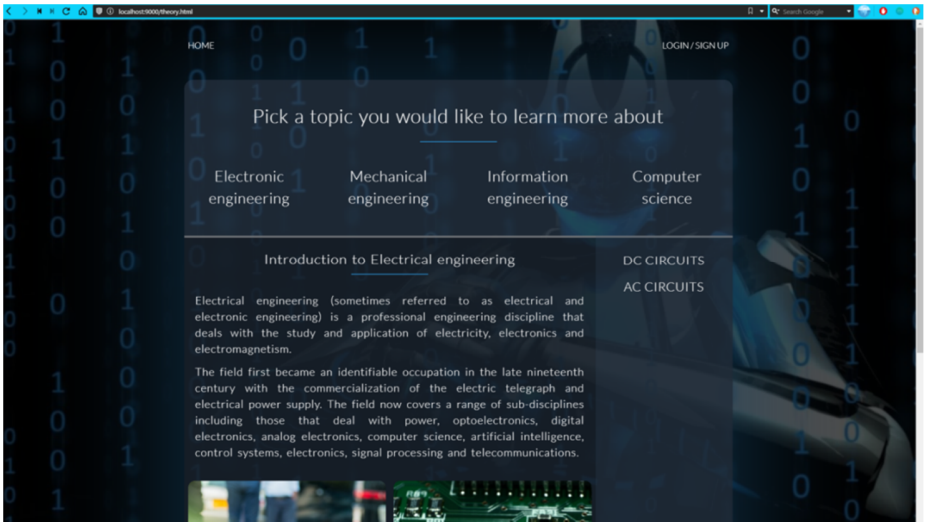
```
state.isSelected = false;
addIdToSection(data.topics);

elements.majorTopics.addEventListener('click', e => {
    //1. Which section is clicked
    const goToTopics = e.target.dataset.major;

    if (goToTopics) {
        const selectedMajorTopic = new MajorTopic(goToTopics);
        //2. Store selected topic into state
        state.selectedMajorTopic = selectedMajorTopic;
        //3. Prepare UI for render
        lectureView.deleteLecture();
        listView.deleteList();
        //4. Render sections
        listView.renderSections(state.selectedMajorTopic.sections);
        //5. Render introduction lecture
        lectureView.renderIntroLecture(data.topics[state.selectedMajorTopic.id])
    }
})

```

Slika 25: Dio glavnog koda

Slika iznad prikazuje dio koda koji je baziran na gornjem dijelu aplikacije. Metoda addEventListener postavlja funkciju koja će se pozvati svaki put kad se navedeni događaj pojavi. To u ovom slučaju znači da nakon odabira lijevom tipkom miša na gornjem dijelu aplikacije, što god se nalazi u vitičastim zagradama će se dogoditi.

Korištenjem konstante goToTopics može se odrediti koja glavna tema je odabrana jer sadrži identifikacijski broj koji je očitan preko HTML atributa data. Pošto ovo vrijedi samo odabirom jedne od glavnih tema korištena je if naredba.

Primjenom izraza new kreiran je novi objekt s ulaznom konstantom goToTopics te je objekt spremljen u konstantu selectedMajortopic. Konstanta selectedMajortopic je potom spremljena u state pod konstantom selectedMajortopic. Nakon toga je potrebno obrisati cijeli donji dio aplikacije (što uključuje uvod i moguće teme) te prikazati uvod i moguće teme što je prikazano na slici ispod. U slučaju da se prvo prikaže uvod i moguće teme pa nakon toga se to obriše, poslije drugog klika na glavnu temu u gornjem dijelu aplikacije će se uvod i moguće teme prikazati ispod prijašnjih prikazanih mogućih tema i uvoda.



Slika 26: Rezultat koda sa slike 25

Sljedeći korak je prikazivanje podteme koja se nalazi na istom mjestu kao i teme. Potrebno je dodavanje gumba za povratak nazad na poziciju prikazanu na slici iznad. Slika ispod prikazuje prvi dio koda koji je odgovoran za donji dio aplikacije. Na sličan način kao i u prošlom dijelu se kreće s metodom addEventListener. Jedina razlika je što je selektirana druga klasa koja označava donji dio aplikacije.

```
elements.topicsList.addEventListener('click', e => {
    // Render list of topics
    if (state.isSelected === false) {
        //1. Which section is clicked
        const goToSection = e.target.dataset.sectionid;

        if (goToSection) {
            const selectedSection = new Section(goToSection);
            //2. Store selected topic into state
            state.selectedSection = selectedSection;
            state.selectedSection.sections.forEach(el => {
                if (state.selectedSection.id === el[0].id) {
                    el.forEach(cur => {
                        //3. Prepare UI for render
                        listView.deleteList();
                        //4. Render subsection title
                        listView.renderSubSecTitle(
                            state.selectedMajorTopic.sections,
                            state.selectedSection.id
                        );
                        //5. Render lectures or Comming Soon
                        cur.title ? listView.renderList(state.selectedSection.id) : listView.renderCommingSoon();
                        //6. Rendet back button
                        listView.renderBackBtn(state.selectedSection.id);
                    })
                }
            })

            return state.isSelected = true;
        }
```

Slika 27: Dio glavnog koda

Varijabla isSelected služi kako bi se znalo što je prikazano na desnoj strani donjeg dijela aplikacije. Može biti u dva stanja: true i false. Ako je u stanju false to znači da su prikazane samo teme, ako je true znači da su prikazane moguće podteme, naslov teme i gumb za povratak nazad.

Procedura kreiranja novog objekta i spremanje konstante u state je ista kao i u prošlom djelu sve do forEach metode. Ova metoda izvršava funkciju jednom za svaki element u nizu. Koja će se metoda koristiti strogo ovisi o tipu podataka. Ovdje se koristi kako bi se prošlo kroz podatke i preko naredbe if našlo koji identifikacijski broj odgovara odabranoj temi.

Nakon toga proces je vrlo sličan kao i u prvom djelu glavnog koda. Obriše se desna strana donjeg dijela aplikacije te se prikažu moguće podteme. Uvjetni ternarni operator je jedini JavaScript operator koji ima tri operanda: uvjet praćen upitnikom, između dvotočke se nalazi izraz koji se izvršava ako je uvjet ispunjen i ako nije ispunjen. Pomoću forEach metode vrši se pretraga da li postoji varijabla title. Ako postoji prikazat će se moguće podteme, a u slučaju da ne postoji prikazat će se poruka da podteme uskoro dolaze.

Što se tiče prikazivanja još je samo ostalo prikazati gumb za povratak nazad na prijašnje teme. Ovaj dio glavnog koda završava tako da se vrati varijabla isSelected s vrijednošću true.

```
79      // Render lecture
80      } else if (state.isSelected === true) {
81          //1. Which is clicked
82          const selectedLecture = e.target.dataset.topicid;
83          const backBtn = e.target.dataset.backbtnid;
84
85          if (selectedLecture) {
86              const lectureObj = new Lecture(selectedLecture);
87              //2. Prepare UI for render
88              lectureView.deleteLecture();
89              //3. Render lecture
90              lectureView.renderLecture(
91                  lectureObj.id,
92                  data.topics[state.selectedMajorTopic.id].sections
93              );
94          }
95
96          if (backBtn) {
97              //1. Prepare UI for render
98              lectureView.deleteLecture();
99              listView.deleteList();
100             //2. Render sections
101             listView.renderSections(state.selectedMajorTopic.sections);
102             //3. Render introduction lecture
103             lectureView.renderIntroLecture(data.topics[state.selectedMajorTopic.id]);
104
105             return state.isSelected = false;
106         }
107     }
108 })
```

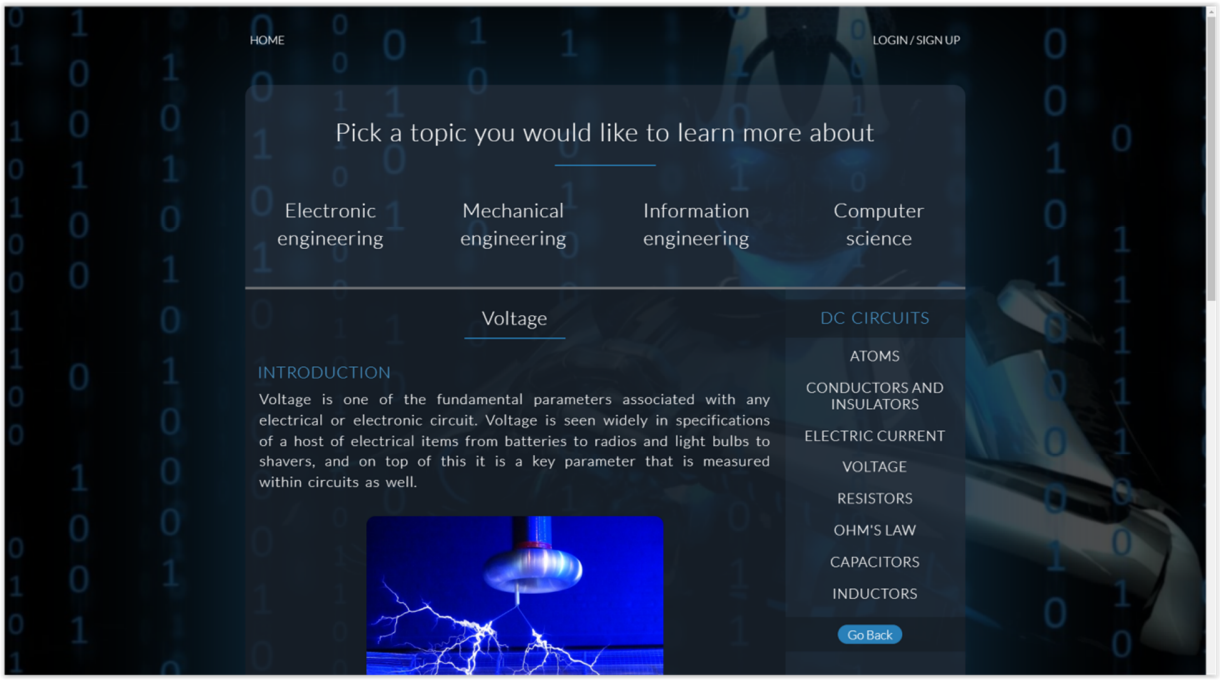
Slike 28: Dio glavnog koda

Na slici iznad je prikazan drugi dio koda zaduženog za obje strane donjeg dijela aplikacije. Ovdje su dvije mogućnosti koje je moguće odabrati: neku od podtema i gumb za povratak nazad. Stoga su dvije if naredbe stavljene za ta dva slučaja.

Prva if naredba se odnosi na prikaz sadržaja izabrane podteme. Postupak je sličan kao i u prethodnom dijelu koda. Kreirao se novi objekt korištenjem funkcije new s ulaznom vrijednošću selectedLecture te je spremljen u konstantu lectureObj. Nakon prethodno odrađenih aktivnosti se obriše sadržaj podteme u slučaju da je postojao i prikaže novi sadržaj.

Druga if naredba se odnosi na gumb za povratak nazad na početak gdje su prikazane moguće teme i uvod u glavnu temu. U ovom slučaju je postupak nešto duži ali i dalje vrlo sličan postupku u prvoj if naredbi. Potrebno je obrisati sadržaj podteme i moguće podteme te slijedi prikazivanje tema i uvoda od prije odabrane glavne teme. Navedeni dio završava vraćanjem varijable isSelected s vrijednošću false.

Na slici ispod je prikazan rezultat aplikacije u kojoj je odabrana glavna tema „Electronic engineering", tema „DC CIRCUITS" te podtema „Voltage".



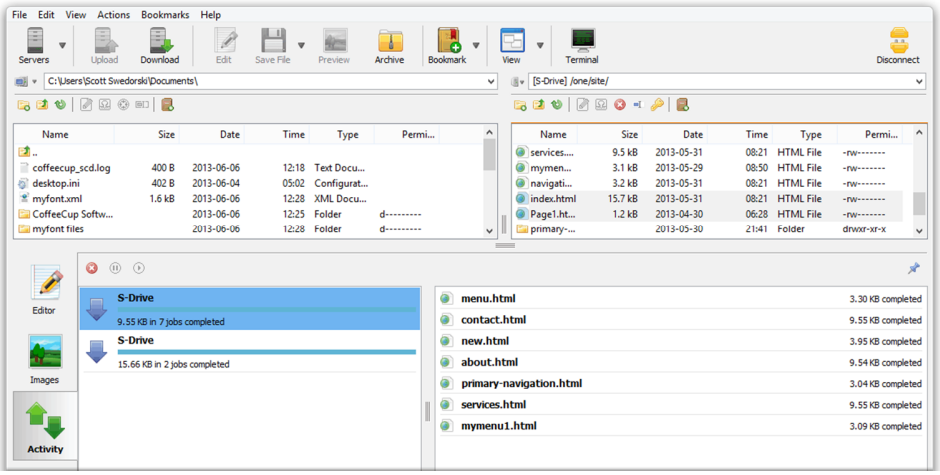
Slika 29: Rezultat kombinacije HTML-a, CSS-a i programskog jezika JavaScript

7. OBJAVLJIVANJE WEB STRANICE

Nakon što se napiše kod i organiziraju datoteke koje čine web stranicu, potrebno je objaviti stranice na Internetu. Objavljivanje web stranice je opširna tema, stoga će ovdje biti objašnjena tri pristupa [78].

7. 1. Odabir domene i prostora na serveru

Web hosting je iznajmljeni prostor na web serveru hosting tvrtke [88]. Hosting tvrtka je tvrtka koja pruža uslugu web hostinga što omogućuje pojedincima i organizacijama da objave svoju web stranicu [87]. Naziv domene jedinstvena je adresa na kojoj korisnici pristupaju web stranici te se domena iznajmljuje na godišnjoj bazi [88].



Slika 30: Primjer besplatnog FTP programa: Free FTP [86]

Za rad s podacima na web serveru potreban je program za prijenos datoteka na web server da bi se datoteke poslale na server. File Transfer Protocol (FTP) programi se uvelike razlikuju, ali općenito se koriste za povezivanje s web serverom koristeći detalje koje pruža hosting tvrtka (korisničko ime, lozinka, ime računala). FTP program prikazuje lokalne datoteke i datoteke web servera u dva prozora i omogućava transfer datoteka između njih [88].

7. 2. Korištenje dodatnih alata

GitHub je servis za kreiranje i održavanje koda aplikacija i projekata. Mnoga su spremišta koda javno dostupna te tako programeri uče jedni od drugih i nadograđuju svoj rad [89]. GitHub ima značajku nazvanu GitHub Pages, koja omogućuje jednostavno i besplatno objavljivanje web stranice s više datoteka na adresi yourusername.github.io. Za objavu statičkih web stranica može se koristiti servis Google App Engine. Statičke web stranice mogu sadržavati tehnologije na strani klijenta kao što su HTML, CSS i JavaScript. Gostovanje statičke web stranice korištenjem App Engine-u je besplatno [88].

7. 3. Alati za online uređivanje koda

Većina programera ovisili su o offline programskim alatima za kodiranje pošto su se prije web preglednici koristili većinom za prikaz web stranica. Prije nekoliko godina pojavili su se web IDE (engl. Integrated Development Environments) kao i uređivači koda te ovih dana postepeno napreduju u odnosu na offline opcije. IDE je softver namijenjen programerima i inženjerima u izradi softvera ili web stranice [90].

Postoji nekoliko web aplikacija koje oponašaju okruženje za razvoj web stranice, omogućujući unos HTML-a, CSS-a i JavaScripta, a zatim rezultat koda prikazan je kao web stranica u pregledniku [88].

8. ZAKLJUČAK

U početcima kada su web stranice postale dostupnije većem broju korisnika, one su korištene za prezentaciju osoba, poduzeća, ustanova te ostale namjene. Kako na globalnoj razini Internet postaje sve dostupniji, tako i web stranice postaju sve kompleksnije u smislu da sve više teže prema online uslugama i interaktivnim prezentacijama. Također danas se velik broj različitih usluga može obaviti korištenjem Interneta, korištenjem web aplikacija poput Internet bankarstva ili sustava e-građani. Zbog zadovoljenja rastućih potreba za različitim uslugama raste potreba za web programerima posebno u 21. stoljeću i može biti vrlo privlačna karijera. Neke od prednosti je mogućnost rada od kuće, raditi samostalno kao freelance (samozaposlena osoba koja sama pronalazi poslove) web programer ili osnovati vlastitu tvrtku. U slučaju da to nisu interesi ili ciljevi i dalje postoji mogućnost rada u profitabilnoj tehnološkoj industriji. Kao i u većini slučajeva, put do toga je vrlo težak pogotovo ako ne postoje nikakva prijašnja znanja. Na prvi pogled to može izgledati vrlo zahtjevno, zastrašujuće ili čak obeshrabrujuće. Kako bi se dobila šira slika o Internetu potrebno je znati osnovne pojmove. Na ovaj način je puno lakše razumjeti kako i zašto neke stvari funkcioniraju što može pomoći pri rješavanju problema. Prije nego što se krene s daljnjim razvojem web stranice, potrebno je odlučiti koju svrhu bi ta web stranica imala i na koji način bi se to najefikasnije postiglo. U ovom radu je bio cilj razviti sustav za elektroničko učenje iz robotike koji bi omogućio prikaz određenih tema. Učinkovitost stjecanje znanja pomoću sustava za elektroničko učenje ovisi o samom sustavu, odnosno kako je osmišljen te o osobnostima korisnika pošto ovaj način učenje neće pristajati svakom korisniku. Korisnicima kojima pristaje takav način učenja, može im pomoći izgraditi potrebne vještine za postizanje osobnih ciljeva. Cilj elektroničkog učenja nije izbaciti standardni oblik učenja već da bude opcija za dodatno stjecanje znanja. Kako bi se postigao zadani cilj, potrebno je obratiti pozornost na web dizajn. Boja, kontrast i kompozicija možda nije ono što ljudi svjesno primjećuju ili o čemu razmišljaju, ali ipak to često može biti razlog zašto ljudi ne steknu dobar prvi dojam o samoj stranici. Stoga je potrebno izdvojiti dovoljno vremena istraživajući karakteristike dizajna kako bi stranica prenijela poruku koju želi projicirati. Ono što se primjećuje je raspored stranice i moguća interakcija s elementima. Raspored stranice bi trebao biti takav da je jednostavno i logično doći do željenih informacija, a interakcija taj proces čini zanimljivim. Internet i tehnologije za izradu

web stranica su znatno napredovale u zadnjih par desetljeća, stoga je relativno lako doći do osnovnih efekata koji mogu pomoći pri povezivanju s publikom. Isto tako postoji i druga strana napredne tehnologije; razvoj mobilnih uređaja koji imaju pristup Internetu je pokrenuo novi pristup izradi web stranica koje moraju biti optimizirane za takve uređaje. Kako zahtjevi za web stranice budu bili sve veći i složeniji, tako će se i načini izrade web stranica morati prilagođavati što znači neprestano učenje za web programere kako bi mogli pratiti taj tempo.

9. LITERATURA

[1] Ian Sample: „What is the internet? 13 key questions answered“ , https://www.theguardian.com/technology/2018/oct/22/what-is-the-internet-13-key-questions-answered, pristupljeno 10.5.2020.

[2] wikipedia: „Internet“ , https://en.wikipedia.org/wiki/Internet, pristupljeno 10.5.2020.

[3] Giovanni Navarria: „How the Internet was born: A stuttered hello “ , https://theconversation.com/how-the-internet-was-born-a-stuttered-hello-67903, pristupljeno 15.7.2020.

[4] Giovanni Navarria: „How the Internet was born: The network begins to take shape “, https://theconversation.com/how-the-internet-was-born-the-network-begins-to-take-shape-67904, pristupljeno 10.5.2020.

[5] Giovanni Navarria: „How the Internet was born: the ARPANET Comes to Life “, https://theconversation.com/how-the-internet-was-born-the-arpanet-comes-to-life-68062, pristupljeno 10.5.2020.

[6] Giovanni Navarria: „How the Internet was born: from the ARPANET to the Internet“, https://theconversation.com/how-the-internet-was-born-from-the-arpanet-to-the-internet-68072, pristupljeno 10.5.2020.

[7] Computer Hope: „Server “, https://www.computerhope.com/jargon/s/server.htm, pristupljeno 10.5.2020.

[8] Computer Hope: „Client “, https://www.computerhope.com/jargon/c/client.htm, pristupljeno 10.5.2020.

[9] Zubin Pratap: „System Design Interview Questions – Concepts You Should Know“, https://www.freecodecamp.org/news/systems-design-for-interviews/, pristupljeno 10.5.2020.

[10] cloudflare: „What is the Internet Protocol? “, https://www.cloudflare.com/learning/network-layer/internet-protocol/, pristupljeno 10.5.2020.

[11] wikipedia: „IP address“ , https://en.wikipedia.org/wiki/IP_address, pristupljeno 10.5.2020.

[12] geeksforgeeks: „Difference between Private and Public IP addresses“, https://www.geeksforgeeks.org/difference-between-private-and-public-ip-addresses/, pristupljeno 10.5.2020.

[13] Computer Hope: „Domain “, https://www.computerhope.com/jargon/d/domain.htm, pristupljeno 10.5.2020.

[14] Computer Hope: „DNS “, https://www.computerhope.com/jargon/d/dns.htm, pristupljeno 10.5.2020.

[15] NortonLifeLock: „What are cookies? “, https://us.norton.com/internetsecurity-how-to-what-are-cookies.html, pristupljeno 10.5.2020.

[16] Kaspersky: „What Are Cookies? “, https://www.kaspersky.com/resource-center/definitions/cookies, pristupljeno 10.5.2020.

[17] cloudflare: „What Is The OSI Model? “, https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/, pristupljeno 10.5.2020.

[18] freeccnastudyguide: „1-3 OSI Reference Model “, https://www.freeccnastudyguide.com/study-guides/ccna/ch1/1-3-osi-reference-model/, pristupljeno 10.5.2020.

[19] Microsoft: „ Windows Network Architecture and the OSI Model“, https://docs.microsoft.com/en-us/windows-hardware/drivers/network/windows-network-architecture-and-the-osi-model, pristupljeno 10.5.2020.

[20] codeguru: „ Introduction to Network Protocols“, https://www.codeguru.com/cpp/sample_chapter/article.php/c12219/Introduction-to-Network-Protocols.htm, pristupljeno 10.5.2020.

[21] Jonathan Strickland: „How does the Internet work? “, https://computer.howstuffworks.com/internet/basics/internet.htm, pristupljeno 10.5.2020.

[22] talentlms: „What is e-learning? Is it important in education? “, https://www.talentlms.com/ebook/elearning/what-is-elearning, pristupljeno 10.5.2020.

[23] talentlms: „Learning vs. Training, what is the difference?“, https://www.talentlms.com/ebook/elearning/learning-vs-training, pristupljeno 10.5.2020.

[24] Caroline Lawless: „What is eLearning?“, https://www.learnupon.com/blog/what-is-elearning/, pristupljeno 10.5.2020.

[25] talentlms: „The history of e-learning“, https://www.talentlms.com/ebook/elearning/history-of-elearning, pristupljeno 10. 5. 2020.

[26] YourTrainingEdge: „Learning vs. Training, What is the Difference?“, https://www.yourtrainingedge.com/learning-vs-training-what-is-the-difference/, pristupljeno 10.5.2020.

[27] talentlms: „The benefits and drawbacks of online learning “, https://www.talentlms.com/ebook/elearning/benefits-and-drawbacks-of-online-learning, pristupljeno 10.5.2020.

[28] Sander Tamm: „What is E-Learning?“, https://e-student.org/what-is-e-learning/, pristupljeno 10.5.2020.

[29] Ilona Hetsevich: „Advantages and Disadvantages of E-Learning Technologies for Students“, https://www.joomlalms.com/blog/guest-posts/elearning-advantages-disadvantages.html, pristupljeno 10.5.2020.

[30] treefrog: „What is Web Design?“, https://www.treefrog.ca/What-is-Web-Design/, pristupljeno 10.5.2020.

[31] interaction-design: „What is Web Design?“, https://www.interaction-design.org/literature/topics/web-design, pristupljeno 10.5.2020.

[32] Marc Schenker: „7 SKILLS EVERY WEB DESIGNER MUST MASTER“, https://www.webdesignerdepot.com/2015/03/7-skills-every-web-designer-must-master/, pristupljeno 10.5.2020.

[33] DecoArt: „Color Theory Basics: The Color Wheel “, https://decoart.com/blog/article/318/color_theory_basics_the_color_wheel#:~:text=The basic color wheel is a collection of 12 colors, pristupljeno 10.5.2020.

[34] Kris Decker: „The fundamentals of understanding color theory“, https://99designs.com/blog/tips/the-7-step-guide-to-understanding-color-theory/, pristupljeno 10.5.2020.

[35] Jessica Stewart: „Learn How Color Theory Can Push Your Creativity to the Next Level“, https://mymodernmet.com/basic-color-theory/, pristupljeno 10.5.2020.

[36] Brad Litwin: „The 5 Types of Contrast in Web Design“, https://www.a2hosting.com/blog/5-types-of-contrast-in-web-design/, pristupljeno 10.5.2020.

[37] John O’Nolan: „FULLY UNDERSTANDING CONTRAST IN DESIGN“, https://www.webdesignerdepot.com/2010/09/fully-understanding-contrast-in-design/, pristupljeno 10.5.2020.

[38] Shavaughn Haack: „A Journey Through Beautiful Typography In Web Design“, https://www.smashingmagazine.com/2013/08/beautiful-typography-web-design/, pristupljeno 10.5.2020.

[39] Suzanne Scacca: „A Reference Guide For Typography In Mobile Web Design “, https://www.smashingmagazine.com/2018/06/reference-guide-typography-mobile-web-design/, pristupljeno 10.5.2020.

[40] Tommy Walker: „The Effect of Typography on User Experience and Conversions “, https://cxl.com/blog/the-effects-of-typography-on-user-experience-conversions/, pristupljeno 10.5.2020.

[41] Lukas Majzlan: „Gas compressing web“, https://dribbble.com/shots/1765503-Gas-compressing-web/attachments/287788, pristupljeno 10.5.2020.

[42] AliceL: „Contrast & Repetition: Setting Elements Apart “, https://www.webdesign.org/contrast-repetition-setting-elements-apart.22517.html, pristupljeno 10.5.2020.

[43] Ryan Boudreaux: „ Effective design principles for web designers: Repetition“, https://www.techrepublic.com/blog/web-designer/effective-design-principles-for-web-designers-repetition/, pristupljeno 10.5.2020.

[44] Marc Schenker: „HOW TO USE BALANCE IN WEB DESIGN“, https://www.webdesignerdepot.com/2015/10/how-to-use-balance-in-web-design/, pristupljeno 10.5.2020.

[45] Visual Hierarchy: „Balance In Web Design And Why It Is Important “, https://visualhierarchy.co/blog/balance-in-web-design-and-why-it-is-important/, pristupljeno 10.5.2020.

[46] Mads Soegaard: „Visual Hierarchy: Organizing content to follow natural eye movement patterns“, https://www.interaction-design.org/literature/article/visual-hierarchy-organizing-content-to-follow-natural-eye-movement-patterns, pristupljeno 10.5.2020.

[47] Rob Bowen: „WHY NEGATIVE IS POSITIVE IN WEB DESIGN“, https://www.webdesignerdepot.com/2013/12/why-negative-is-positive-in-web-design/, pristupljeno 10.5.2020.

[48] Anastasia Diachenko: „Effective Use of Negative Space in Web Design“, https://speckyboy.com/negative-space-web-design/, pristupljeno 10.5.2020.

[49] WebFX: „Why Is Web Design Important?“, https://www.webfx.com/web-design/why-is-web-design-important.html, pristupljeno 10.5.2020.

[50] WebDesignDev: „5 Essential Skills Every Successful Web Designer Needs“, https://www.webdesigndev.com/5-essential-skills-every-successful-web-designer-needs/, pristupljeno 10.5.2020.

[51] Oxxy: „8 Essential Web Design Skills You Need to Excel Right Now“, https://www.oxxy.com/blog/en/web-design-skills/, pristupljeno 10.5.2020.

[52] Kinsta: „What Is a Content Management System (CMS)? “, https://kinsta.com/knowledgebase/content-management-system/, pristupljeno 10.5.2020.

[53] Tessa Roberts: „What is a Content Management System and How Does it Work?“, https://www.bloomreach.com/en/blog/2019/08/content-management-system.html, pristupljeno 10.5.2020.

[54] Savaram Ravindra: „Content Management Systems Explained“, https://www.relevance.com/content-management-systems-explained/, pristupljeno 10.5.2020.

[55] Umbraco: „What is a CMS? “, https://umbraco.com/about-us/umbraco-dictionary/cms/#pros-and-cons-of-a-cms, pristupljeno 10.5.2020.

[56] wikipedia: „WordPress“, https://en.wikipedia.org/wiki/WordPress, pristupljeno 10.5.2020.

[57] WordPress: „Meet WordPress“, https://wordpress.org/#:~:text=Beautiful designs, powerful features, and the freedom to build anything, pristupljeno 10.5.2020.

[58] drupalizeme: „What Is Drupal? “, https://drupalize.me/what-is-drupal, pristupljeno 10.5.2020.

[59] Drupal: „Credit Due Admin Theme “, https://www.drupal.org/project/credit_due#:~:text=This project is not covered by Drupal’s security advisory policy.,pristupljeno 10.5.2020.

[60] rockettheme: „What is Joomla? “, https://rockettheme.com/docs/joomla/platform, pristupljeno 10.5.2020.

[61] Arvind Chauhan: „8 most exciting Joomla 4 new features“, https://www.joomlart.com/blog/8-most-excited-joomla-4-new-features#:~:text=New back-end navigation system on Joomla 4, pristupljeno 10.5.2020.

[62] marionletendart: „What is Web Development?“, https://blog.openclassrooms.com/en/2018/03/28/web-development-definition/, pristupljeno 10.5.2020.

[63] BitDegree: „What Is A Web Developer: Understanding What Is Web Development“, https://www.bitdegree.org/tutorials/what-is-a-web-developer/, pristupljeno 10.5.2020.

[64] Michael Wales: „3 Web Dev Careers Decoded: Front-End vs Back-End vs Full Stack“, https://blog.udacity.com/2014/12/front-end-vs-back-end-vs-full-stack-web-developers.html, pristupljeno 10.5.2020.

[65] Cody Lindley: „What Is a Front-End Developer?“, https://frontendmasters.com/books/front-end-handbook/2018/what-is-a-FD.html, pristupljeno 10.5.2020.

[66] w3schools.com: „Web Development Roadmaps “, https://www.w3schools.com/whatis/#:~:text=Chart created by GitHub user Kamranahmedse: https://github.com/kamranahmedse, pristupljeno 10.5.2020.

[67] Priyesh Patel: „What exactly is Node.js“, https://www.freecodecamp.org/news/what-exactly-is-node-js-ae36e97449f5/, pristupljeno 10.5.2020.

[68] stackshare: „npm“, https://stackshare.io/npm, pristupljeno 10.5.2020.

[69] Web Programiranje: „npm & yarn osnove“, https://www.webprogramiranje.org/npm-yarn-osnove/, pristupljeno 10.5.2020.

[70] squarespace: „What is JSON“, https://developers.squarespace.com/what-is-json, pristupljeno 10.5.2020.

[71] Node.js: „What is the file `package.json`?“, https://nodejs.org/en/knowledge/getting-started/npm/what-is-the-file-package-json/, pristupljeno 10.5.2020.

[72] Ryan Glover: „What are JavaScript packages and dependencies? “, https://cleverbeagle.com/blog/articles/what-are-javascript-packages-and-dependencies#:~:text=A dependency is some third,application cannot function without it., pristupljeno 10.5.2020.

[73] Web Programiranje: „Webpack (osnove)“, https://www.webprogramiranje.org/webpack-osnove/, pristupljeno 10.5.2020.

[74] Ciel: „What is Webpack and why should I care? [Part 1][Introduction] “, https://medium.com/the-self-taught-programmer/what-is-webpack-and-why-should-i-care-part-1-introduction-ca4da7d0d8dc, pristupljeno 10.5.2020

[75] Computer Hope: „HTML “, https://www.computerhope.com/jargon/h/html.htm, pristupljeno 10.5.2020.

[76] MDN web docs: „ Getting started with HTML“, https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Getting_started, pristupljeno 10.5.2020

[77] MDN web docs: „ : The Content Division element“, https://developer.mozilla.org/en-US/docs/Web/HTML/Element/div, pristupljeno 10.5.2020.

[78] MDN web docs: „<ul>: The Unordered List element“, https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ul, pristupljeno 10.5.2020.

[79] MDN web docs: „<li>“, https://developer.mozilla.org/en-US/docs/Web/HTML/Element/li, pristupljeno 10.5.2020.

[80] Computer Hope: „HTML validator “, https://www.computerhope.com/jargon/h/html-validator.htm, pristupljeno 10.5.2020

[81] MDN web docs: „What is CSS?“, https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS, pristupljeno 10.5.2020.

[82] MDN web docs: „Selector (CSS)“, https://developer.mozilla.org/en-US/docs/Glossary/CSS_Selector, pristupljeno 10.5.2020.

[83] MDN web docs: „Pseudo-classes“, https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes, pristupljeno 10.5.2020.

[84] MDN web docs: „What is JavaScript?“, https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript, pristupljeno 10.5.2020.

[85] BitDegree: „What Is JavaScript Used For And Why You Should Learn It“, https://www.bitdegree.org/tutorials/what-is-javascript-used-for/, pristupljeno 10.5.2020.

[86] CoffeeCup: „FREE FTP“, https://www.coffeecup.com/free-ftp/#:~:text=Fast and Efficient File Transfers., pristupljeno 10.5.2020.

[87] wikipedia: „Web hosting service“, https://en.wikipedia.org/wiki/Web_hosting_service, pristupljeno 10.5.2020.

[88] MDN web docs: „Publishing your website“, https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/Publishing_your_website, pristupljeno 10.5.2020.

[89] Khan Academy: „Hosting your website on Github“, https://www.khanacademy.org/computing/computer-programming/html-css/web-development-tools/a/hosting-your-website-on-github, pristupljeno 10.5.2020.

[90] Devrix: „Web IDEs: The Future of Coding“, https://devrix.com/tutorial/web-ides-future-coding/, pristupljeno 10.5.2020.

10. POPIS SLIKA