Izrada responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava

Bilić, Branimir

Sveučilište
Sjever

Završni rad broj 534/MM/2017

IZRADA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA

Student

Branimir Bilić, 5515/601

Varaždin, rujan 2018.

Sveučilište
Sjever

Odjel za multimediju, oblikovanje i primjenu

Završni rad broj 534/MM/2017

IZRADA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA

Student

Branimir Bilić, 5515/601

Mentor

Izv. prof. dr. sc. Mario Tomiša

Varaždin, rujan 2018.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL	Odjel za multimediju, oblikovanje i primjenu		
PRISTUPNIK	Branimir Bilić	MATIČNI BROJ	5515/601
DATUM	01.06.2017.	KOLEGIJ	Web dizajn
NASLOV RADA	Izrada responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava		
NASLOV RADA NA ENGL. JEZIKU	Creating responsive web site using Bootstrap framework and Wordpress CMS system		
MENTOR	dr. sc. Mario Tomiša	ZVANJE	Izvanredni profesor

ČLANOVI POVJERENSTVA

1. mr.sc. Vladimir Stanisavljević, v.pred. - predsjednik
2. doc.dr.sc. Dean Valdec - član
3. izv.prof.dr.sc. Mario Tomiša - mentor
4. dr.sc. Robert Logožar, v. pred. - zamjenski član
5.

Zadatak završnog rada

BROJ 534/MM/2017

OPIS

Bootstrap je set HTML, CSS i JS komponenti koje su namijenjene brzoj izradi web stranica, odnosno grafičkih korisničkih sučelja web orijentiranih projekata. Takva kolekcija unaprijed pripremljenih kodova i alata naziva se „radni okvir", odnosno „framework". Bootstrap je „frontend framework" koji sadrži HTML i CSS kodove za brzu ugradnju raznih UI (eng. User-Interface) elemenata poput obrazaca, navigacije, galerije multimedijskih sadržaja i naprednih dinamičnih komponenti temeljenih na JavaScript programskom jeziku. Inicijalnu verziju izradili su Mark Otto i Jacob Thorton 2010. godine, dok su radili kao dizajneri u tvrtci Twitter Inc. Alat je inicijalno namijenjen očuvanju dosljednosti u dizajnu korisničkih sučelja, no s vremenom je postao slobodno dostupan korisnicima kao „open source" softver. Jednostavnom modifikacijom CSS koda moguće je brzo izmijeniti grafiku HTML elemenata, zbog čega je Bootstrap postao najpopularniji „frontend framework" za web dizajn. WordPress je trenutno najpopularniji CMS (eng. Content Management System). Inicijalnu verziju izradio je Matt Mullenweg, 27. svibnja 2003. godine, s ciljem razvoja praktičnog sustava koji bi olakšao aktivnosti vezane za pisanje i objavu sadržaja na webu. Primarno je namijenjen za objavu osobnih blogova, no s vremenom se razvio u praktičan sustav za upravljanje sadržajem. Sustav je temeljen na PHP programskom jeziku i MySQL bazi podataka, a „frontend" sustav moguće je nadograđivati vlastitim grafičkim predlošcima temeljenim na HTML, CSS i JavaScript jezicima. Glavne odlike WordPress sustava su jednostavnost korištenja te veliki izbor dodataka (eng. plugins) koji omogućuju fleksibilnost te prilagodbu sustava specifičnim potrebama korisnika. Stoga, kombinacijom WordPress sustava i Bootstrap radnog okvira moguće je u kratkom vremenskom roku izraditi kvalitetne i moderne web stranice ili aplikacije. Ovaj završni rad će prikazati izradu responzivnog web mjesta pomoću navedenih alata i tehnologija. Prvi korak je izrada potrebnih web stranica pomoću Bootstrap radnog okvira, a potom njihova implementacija u WordPress CMS sustava. U procesu planiranja projekta, odabrat će se potrebni dodaci i proširenja za WordPress sustav te razraditi plan realizacije web mjesta, s ciljem ostvarenja optimalnog odnosa uloženih resursa i konačne kvalitete.
U radu je potrebno:
- Objasniti temeljne pojmove iz područja web dizajna, relevantne za realizaciju praktičnog dijela završnog zadatka: responzivno web mjesto, CMS sustav, web UI komponente, frontend teme, backend panel, CRUD funkcionalnosti
- Objasniti frontend i backend tehnologije u kontekstu web dizajna
- Opisati Bootstrap radni okvir i WordPress CMS sustav te pripadajuće komponente i proširenja
- Prikazati plan realizacije projekta – svrhu, cilj, ciljanu publiku, razradu navigacije, potrebna umijeća, resurse i alate, s naglaskom na kasnije održavanje i nadogradnju sustava
- Prikazati praktičnu realizaciju responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava
- Odrediti prednosti i nedostatke prikazane metode realizacije responzivnog web mjesta u odnosu na klasičan pristup

ZADATAK URUČEN 07.07.2017. POTPIS MENTORA

REPUBLIKA HRVATSKA SVEUČILIŠTE SJEVER

Predgovor

Tema ovog završnog rada je responzivni web dizajn (engl. RWD – Responsive web design) koji podrazumijeva mogućnost prilagodbe web stranice veličini ekrana uređaja na kojem se pregledava, pri čemu je širina zaslona relevantna dimenzija. Također, spominje se kao tehnika kojom se izrađuju web stranice ili web aplikacije, koje se prilagođavaju različitim dimenzijama uređaja ili web preglednika. Nekada davno bilo je dovoljno razviti samo jednu verziju stranice, onu za pregledavanje na zaslonu *desktop* računala. S razvojem mobilnih uređaja, koji su postepeno počeli dobivati pristup internetu, počele se su razvijati web stranice koje se prilagođavaju veličini uređaja na kojim ih korisnik pregledava. Svjetski poznati informativni portal, Mashable, 2013. godinu proglasio je godinom responzivnog web dizajna[1].

Ovaj završni rad prikazuje radnje koje olakšavaju dnevne zadatke web dizajnera u smislu ubrzanja procesa izrade statičnih web stranica i implementacije istih u besplatni CMS sustav, čime se ostvaruje cjeloviti web proizvod.

[1] Cashmore, Pete: „Why 2013 Is the Year of Responsive Web Design“,
URL: https://mashable.com/2012/12/11/responsive-web-design/#vfChzOu0isqZ

SAŽETAK

Danas su mobilni uređaji sve prisutniji kao platforma za pristup internetu, stoga je neophodno uzeti u obzir kako web stranice moraju biti prilagođene za prikaz na tim uređajima. Također, važno je obratiti posebnu pažnju na dostupne tehnologije i koncepte izrade stranica prilagođenih mobilnim uređajima, kao i njihove prednosti i nedostatke. Iako je danas još uvijek velika većina web stranica prilagođena isključivo za prikaz na stolnim računalima, zahtjevi tržišta nameću potrebu da web stranice prilagođene mobilnim uređajima postanu standard na webu. Postoje različiti koncepti izrade web stranica prilagođenih mobilnim uređajima kao što su fluidan, responzivan ili adaptivan dizajn, te je neophodno poznavanje osnovnih principa na koje se oni oslanjaju kako bi se mogli uspješno implementirati u web stranice. Tematika rada usmjerena je na Bootstrap radni okvir i njegove komponente i proširenja, WordPress CMS sustav te izradu responzivnog web mjesta pomoću istih. U teoretskom djelu rada opisane su web tehnike kao što su HTML, CSS te JS kao standardi na web stranicama. Praktični dio rada prikazuje realizaciju responzivnog web mjesta pomoću Bootstrap radnog okvira i njegovu ugradnju u WordPress CMS sustav, pri čemu je naglasak stavljen na mogućnosti održavanja i nadogradnje sustava te prednosti i nedostatke prikazanih metoda pri izradi responzivnog web mjesta, u odnosu na klasičan pristup.

KLJUČNE RIJEČI: Responzivni dizajn, CMS sustav, front-end teme, Bootstrap, WordPress

ABSTRACT

Today, mobile devices are present as Internet access platforms, and it is necessary to consider how websites will be displayed on these devices. Also, it is important to pay special attention to available technologies and concepts of customizing mobile pages, as well as their strengths and weaknesses. Although today, the vast majority of web pages tailored exclusively to display on desktop computers, market demands the need for mobile content-based websites to become standard on the web. There are different concepts of designing mobile-friendly web pages such as fluid, responsive or adaptive design, and it is essential to know the basic principles they rely on so that they can be successfully implemented in a web site. This paper focuses on the Bootstrap framework and its components and extensions, the WordPress CMS system and the creation of a responsive site using the same. The theoretical part of this paper describes web techniques such as HTML, CSS, JS as standards for web design. The practical part of the article demonstrates the realization of a responsive site by using the Bootstrap framework and linking it to the WordPress CMS system with an emphasis on maintaining and upgrading the system, and the advantages and disadvantages of the methods presented in creating a responsive site compared to a classic approach.

KEYWORDS: Responsive design, CMS system, front-end themes, Bootstrap, WordPress

POPIS KORIŠTENIH KRATICA

RWD (engl. Responsive web designe) – Responzivni web dizajn

HTML (engl. HyperText Markup Language) – Opisni jezik kojim se definira struktura web stranice

JS (engl. JavaScript) – Skriptni programski jezik koji se izvršava na strani korisnika (klijenta)

jQuery –JavaScript biblioteka

CSS (engl. Cascading Style Sheets) – Opisni jezik za opisivanje izgleda web stranice

CMS (engl. Content management system) – Sustav za upravljanje sadržajem

PHP (engl. Personal Home Page ili PHP: Hypertext Preprocessor) – Programski jezik koji se izvršava na strani poslužitelja

SQL (engl. Structured Query Language) – Strukturirani upitni jezik za komunikaciju s bazom podataka

MySQL (engl. My -Structured Query Language) – Standardni jezik sistema za upravljanje bazama podataka, tzv. RDBMS (engl. Relational Database Management System)

GPL (engl. General Public License) – Licenca za slobodan softver

GML (engl. Generalized Markup Language) – Jezik za oblikovanje IBM dokumenata koji opisuje organizacijsku strukturu i dijelove sadržaja dokumenta.

GUI (engl. Graphical User Interface) – Grafičko korisničko sučelje

UI (engl. User Interface) – Korisničko sučelje

UX (engl. User Experience) – Korisničko iskustvo

AJAX (engl. Asynchronous JavaScript And XML) – Tehnika koja omogućuje izradu web aplikacija.

ACF (engl. Advanced Custom Fields) – WordPress plugin za realizaciju naprednih input polja.

SADRŽAJ

1. UVOD 1
2. IZRADA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA 3
2.1. RESPONZIVNI DIZAJN 3
2.2. WEB DIZAJN 5
2.3. RESPONZIVNO WEB MJESTO 7
2.4. FLUIDNA MREŽA 7
3. CMS SUSTAV 9
3.1. PROGRAMSKI JEZICI 9
3.2. BAZE PODATAKA 10
4. WEB UI KOMPONENTE 10
5. FRONTEND I BACKEND TEHNOLOGIJE U KONTEKSTU WEB DIZAJNA 11
5.1. FRONTEND TEME 11
5.2. JEZICI ZA KODIRANJE FRONTEND-a 12
5.3. PREDUVJETI ZA KODIRANJE FRONTEND-a 12
5.4. BACKEND PANEL 12
5.5. CRUD FUNKCIONALNOSTI 14
6. BOOTSTRAP RADNI OKVIR I NJEGOVE KOMPONENTE 15
6.1. HTML 16
6.2. CSS 17
6.3. JAVASCRIPT 18
6.4. BOOTSTRAP MAPA 19
6.5. BOOTSTRAP TIPOGRAFIJA 20
6.6. LISTE I TABELE 20
6.7. FORME 21
6.8. GUMBI I SLIKE 23
6.9. GLYPHICONS 23

6.10.RAZLIKE IZMEĐU STATIČNIH I DINAMIČNIH WEB STRANICA 24
6.11.WIREFRAMING 24
6.12.SKIDANJE I INSTALACIJA BOOTSTRAPA 24
6.13.POSTAVLJANJE RADNE OKOLINE PROJEKTA I KODIRANJE 24
7. WORDPRESS CMS SUSTAV I NJEGOVE KOMPONENTE 26
7.1. NADZORNA PLOČA 27
8. PLAN REALIZACIJE PROJEKTA 30
9. PRAKTIČNA REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA 31
9.1. REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA 31
9.1.1. HEADER 32
9.1.2. HERO IMG ZONA 34
9.1.3. MODAL 35
9.2. REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU WORDPRESS CMS SUSTAVA 43
9.2.1. HEADER 44
9.2.2. FOOTER 45
9.2.3. HOME PAGE 46
9.2.4. HERO IMG 47
9.2.5. ADVANCED CUSTOM FIELDS 49
9.2.6. ACF ADD FIELD OPCIJE 49
9.2.7. POSTAVLJANJE BLOGA 50
10. PREDNOSTI I NEDOSTACI PRIKAZANE METODE REALIZACIJE RESPONZIVNOG WEB MJESTA U ODNOSU NA KLASIČAN PRISTUP 55
11. ZAKLJUČAK 57
LITERATURA 59
POPIS IZVORNIH KODOVA 60
POPIS TABLICA 61
POPIS SLIKA 61

1. UVOD

Bootstrap je *frontend* radni okvir (engl. Framework) koji sadrži besplatnu kolekciju alata za stvaranje web-stranica i web-aplikacija. Uključuje HTML i CSS predloške za gumbe, navigaciju, obrasce te JavaScript proširenja. Početnu verziju izradili su Mark Otto i Jacob Thorton 2010. godine, dok su radili kao dizajneri u tvrtki Twitter Inc. U kolovozu 2011. godine, Bootstrap je objavljen kao otvoreni izvor na GitHubu i brzo postao najpopularniji repozitorij na platformi. Iako je *frontend* okvir bio dobro prihvaćen od početka, popularnost je počela kada je izdana verzija 2, prvo ažuriranje s mogućnošću osjetljivih dizajnerskih mogućnosti. On je značajno pridonio obuhvatu ne samo osnovnih stilova, već i elegantnijih predložaka. Već 2012. godine napravljeno je novo izdanje, sljedeće već 2013. godine, u kojem je opisan mobilni pristup i tada je u njega implementiran dizajn prilagodljiv različitim dimenzijama ekrana (RWD). U listopadu 2014. godine, Mark Otto je najavio da je Bootstrap 4 u razvoju, tek u rujnu 2016. godine, obustavio rad na Bootstrap verziji 3 kako bi oslobodio vrijeme za rad na verziji 4. Preko četiri tisuće promjena izvršeno je do sada na Bootstrap 4 kodnoj bazi. Njegova popularnost i jednostavnost integracije postavila ga je na prvo mjesto i danas je nezaobilazan alat za izradu web stranica. [1, 8]

WordPress je sustav za upravljanje sadržajem (CMS). Razvijen je 2003. godine s ciljem olakšavanja aktivnosti pisanja i objave sadržaja na web-u. Primarno je bio namijenjen za objavu osobnih blogova, ali je dodavanjem funkcionalnosti tijekom godina evoluirao u cjeloviti sustav za upravljanje sadržajem. Sustav je implementiran jezikom PHP, oslanja se na sustav za upravljanje bazom podataka MySQL, a njegovo korištenje i razvoj regulirano je licencom GNU General 6 Public License 2.0. WordPress je započeo samo kao blog sustav, ali je evoluirao u to da se može koristiti kao sustav za upravljanje sadržajem koristeći tisuće dodataka, widgeta i tema. Izrađen je iz želje za elegantnim, dobro arhitekturiranim osobnim izdavačkim sistemom izgrađenim na PHP-u i MySQL-u i licenciranim pod GPL-om. On je novi softver, ali njegovi korijeni i razvoj sežu još iz 2001. godine. [9]

WordPress je najpopularniji CMS u upotrebi danas. Dovoljno pokazuje podatak da 30% svih web stranica na cijelom Internetu budu unutar WordPress sustava, a uz to je i najpopularnija blogerska platforma s preko 60 milijuna web blogova[2].

[2] Wikipedia: „WordPress“
URL:https://en.wikipedia.org/wiki/WordPress

Responzivni web dizajn je tehnika kojom se izrađuju web stranice ili web aplikacije koje se prilagođavaju različitim dimenzijama uređaja ili *Browsera* kojima pristupamo. Primjerice, radna površina stolnog računala prikazuje standardnu verziju strance ali može prikazivati i verziju za široki zaslon (engl. Widescreen), što znači da stranica bude optimizirana za veće zaslone računala drugih. Responzivan dizajn razlikuje se od ostalih metodologija po samoj izradi web stranica. Dok se kod ostalih metodologija prvo kreira verzija stranice za stolna računala koja se zatim postepeno prilagođava mobilnim uređajima, kod responzivnog dizajna se kreće upravo od stranice prilagođene mobilnim uređajima, koja se onda postepeno obogaćuje sadržajem za uređaje koji mogu prikazati veći ili bogatiji sadržaj. Tako korisnici mobilnih uređaja izbjegnu nepotrebno učitavanje sadržaja. Također, responzivni dizajn omogućuje tvrtkama povećanje posjetitelja na web stranicu putem svih vrsta digitalnih uređaja i bolje korisničko iskustvo u odnosu na klasičan pristup. Ono što se smatra velikim nedostatkom u responzivnom dizajnu je što su svim korisnicima dostupni uglavnom jednaki multimedijalni sadržaji bez optimizacije za sve vrste uređaja te izuzetno teška optimizacija tablica. Sve do nedavno, web stranice su se izrađivale za fiksnu širinu zaslona, kao što je npr. 960 piksela, uz pretpostavku kako će svi korisnici dobiti konzistentan prikaz samog sadržaja. Ta pretpostavka bila je dobra kod korištenja prijenosnih i stolnih računala, no danas, kada su web stranice jedan od masovnih medija, a korisnici pristupaju web-u sa svog mobitela, tableta i dr. rezultati nisu zadovoljavajući. Mobiteli uzimaju sve veći udio na tržištu, stoga se spominju kao glavna platforma za pristup internetu. [1]

U ovom radu prikazan je osvrt na tehnologije koje su omogućile izradu web stranica prilagođenih mobilnim uređajima, a kroz kratak pregled metodologije web dizajna prikazane su prednosti i nedostaci pojedinih pristupa izradi web stranica. U praktičnom djelu rada prikazan je plan realizacije projekta s naglaskom na kasnije održavanje i nadogradnju sustav, te praktična realizacija responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava.

2. IZRADA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA

2.1.RESPONZIVNI DIZAJN

Responzivni web dizajn je metoda za optimizaciju sadržaja na web stranici za prikaz na uređajima različitih dimenzija. Ova vrsta dizajna kombinira fluidni dizajn s *media upitima*. Kroz izradu stranice koristi se metodologija fluidnog dizajna, a u samom dizajnu predviđaju se određene prijelomne točke nakon kojih se na pojedine elemente primjenjuje novi niz CSS pravila. Na taj se način prikrivaju nedostaci samog fluidnog dizajna jer se reorganizacijom elemenata unutar stranice može se postići adekvatan prikaz elementa za gotovo svaku veličinu ekrana. [1, 3]

Radna površina stolnog računala ima uobičajenu verziju web stranice, ali može imati i verziju za široki zaslon (engl. Widescreen), dakle stranica je optimizirana za veće zaslone. Također, potrebno je optimizirati izgled za tablete, iskoristivši prednosti njihovog portretnog ili pejzažnog izgleda dok je pomoću mobitela, moguće je ciljati njihovu znatno veću širinu. Radi uspješnije optimizacije, Bootstrap koristi CSS medijske upite za mjerenje širine prozora preglednika, a zatim, pomoću *media upita*, mijenja učitane stilove. Koristeći širinu prozora preglednika, Bootstrap može optimizirati sadržaj pomoću kombinacija omjera ili širina, ali se uglavnom oslanja na minimalnu ili maksimalnu širinu ekrana. U osnovi, Bootstrap podržava pet različitih izgleda, od kojih se svaki oslanja na CSS medijske upite. Najveća mreža (engl. Grid) ima stupce širine 70 piksela, u suprotnosti sa 60 piksela od uobičajenih mreža. Oblik mreže donosi stupce širokim do 42 piksela i kada su uži od toga, svaki stupac prelazi u fluidno stanje, što znači da su stupci vertikalno položeni te svaki stupac predstavlja punu širinu uređaja. U tablici broj 1, opisano je pet izgleda prikaza stranica (engl. Wiewport). [1, 3]

Veličina displaya	Širina preglednika	Širina kolumne	Gutter
Veliki display	1200px i više	70px	30px
Standardni	980px i više	60px	20px
Tableti portret	768px i više	42px	20px
Od mobitela do tableta	767px i manje	Fluidne kolumne nema zadanih širina	
Mobiteli	480px i manje	Fluidne kolumne nema zadanih širina	

Tablica 1. Media upiti za veličine preglednika [1]

Bootstrap je pripremljen za mobilne uređaje (engl. Mobile Frendly) gdje postoje različite veličine ekrana (engl. Display) na kojima se pregledava sadržaj, što se može vidjeti u tablici broj 2.

	Extra mali uređaji (Smartphone <768px)	Mali uređaji (Tableti ≥768px)	Srednje veliki (Desktop računala, laptopi ≥992px)	Veliki uređaji (Desktop računala ≥1200px)
Mreža	Vodoravna			
Širina kontejnera	Auto	540px	720px	960px
Klasa	.col-xs-	.col-sm-	.col-md-	.col-lg-
Broj kolumni	12			
Širina margine	30px (15px sa svake strane kolumne)			
Ugnježđenje	Da			
Ofset	Da			

Tablica 2. Veličine preglednika i klase Bootstrapa [3]

Kako bi stranica bila responzivna, u HTML-u u meta elementima mora se dodati *kod* za izračun veličine ekrana (Izvorni *kod* broj 1).

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Izvorni kod broj 1. HTML kod za izračunavanje širine ekrana

Dalje HTML predstavlja pregledniku (engl. Browser) veličinu ekrana, kako bi preglednik mogao prikazati web sadržaj. U svrhu izrade rasporeda (engl. Layout) potrebno je napraviti * * elemente s klasom *row* u koje treba staviti elemente s klasom veličine stupca (*col-sm-4*) i njome odrediti veličinu koju taj element zauzima u rasporedu (Tablica broj 2). Elementi unutar mreže imaju mogućnost stavljanja jednog u drugog (engl. nesting) dodajući unutar elementa s klasom *row* još jedan element s klasom *row* (Izvorni *kod* broj 2). [1, 4]

```
<div class="row">
  <div class="col-*-*"></div>
  <div class="col-*-*"></div>
</div>
```

Izvorni kod broj 2. Bootstrap klasa row

2.2. WEB DIZAJN

Web dizajn je vrsta dizajna srodna grafičkom dizajnu u mnogim segmentima. U web dizajnu poruka se oblikuje tako da bude prilagođena prikazu na webu, uz web preglednik upotpunjena multimedijalnim sadržajima. Različitost web i grafičkog dizajna je u izvedbi. Web dizajn je specifičan zbog svoje dinamičke strukture gdje se web stranice proširuju i neprestano osvježavaju. Uspješnost komunikacije na Internetu polazi od pretpostavke kako je važno privući posjetitelja i ponuditi mu web stranice koje su jednostavne i ugodne za korištenje. [3,7]

Web dizajn obuhvaća različite vještine i standarde koje se koriste u izradi web stranica. Također služi i za planiranje izrade web stranica, a uključuje informacijsku arhitekturu, korisničko sučelje, strukturu stranica, navigaciju, izgled, boje, fontove i slike i dr. Elementi koji su važni u web dizajnu su dizajn korisničkog sučelja, SEO optimizaciju za web pretraživače, *markup* jezike (HTML, XHTML I XML), te stilske jezike (engl. Style sheet) kao što su CSS i XSL, te programske jezike JavaScript i VBScript te baze podataka tehnologije (MySQL, MSSQL). [16]

Povijest web dizajna seže u 1988. godinu, kada je Tim Berners Lee u sklopu CERN-a utemeljio web kao medij za udaljenu razmjenu informacija[3]. Web dizajn je umjetnost prezentacije sadržaja za krajnjeg korisnika na www-u (engl. World Wide Web-u). [17]

U digitalnom svijetu, web stranice pojavljuju se kao značajan alat koji ne samo da pomaže kompanijama da pronađu svoju ciljanu grupu, već i kako bi privukle nove klijente iz cijelog svijeta. Web stranica je postala sredstvo komunikacije i interakcija za poslovanje. Teško je zamisliti internet bez animiranih slika, različitih stilova tipografije, šarenih pozadinskih slika ili glazbe, a za to je odgovoran grafički dizajn u uskoj povezanost s web dizajnom. Povijest interneta počinje u kasnim godinama 1960. Prvo se strogo koristio za razne potrebe američke vojske, a kasnije su ga počela koristiti pojedina sveučilišta. U to vrijeme, internet je postao specifični alat za razmjenu važnih podataka između računala uz pomoć prijenosnih protokola kontrole (TCP / IP). Tijekom 1969. godine, na internetu se pojavio poseban jezik. To su bili ARPANET i GML i komuniciranje putem računala bilo je nerazumljivo. Godine 1972., Ray Tomlinson stvorio je e-mail koji je munjevitom brzinom promijenio kompletan značaj interneta.

[3] Wikipedia: „Web design“
URL: https://en.wikipedia.org/wiki/Web_design

U početku je web dizajn minimalno napredovao, no pojavom HTML – a, postao je složeniji i fleksibilniji. Pojavom CSS-a i tehnologija koje se izvršavaju na strani poslužitelja web dizajn doživljava nagli razvoj. Godine 1993., prvi internet pretraživač, Mozaik - omogućio je korisnicima da pretražuju internet u vizualnom smislu i približio korištenje web tehnologije široj javnosti. Tvorcem je spomenut Nacionalni centar za super računalne aplikacije (engl. National Center for Supercomputing Applications, NCSA). Ovaj *browser*, omogućio je korisnicima da pregledavaju tekstove i grafički dio, no bio je prilično ograničen. Godine 1994., postavljeni su standardi i smjernice budućeg razvoja HTML-a, kako bi se omogućio i dinamički sadržaj na webu. Nedugo nakon toga, 1995. godine, Microsoft je objavio novu verziju operativnog sistema, Windows 95, koji je sadržavao potpuno novo korisničko sučelje i preglednik na Windows platformi: Microsoft Internet Explorer, koji je postao veoma popularan širom svijeta. Nekoliko mjeseci kasnije, Atavista, prvi višejezični pretraživač i Amazon. com, jedna od prvih internet kompanija koja omogućava elektronske transakcije, doprinjele su rasprostranjenosti web-a (engl. World Wide Web). Tehnologije baza podataka, *server-side* programski jezici (PHP, ASP. Net, JSP, ColdFusion, itd) i novi standardi u dizajnu kao što su CSS, znatno su promijenili web i pružili veće mogućnosti za web dizajnere i web programere. [13, 14, 17]

Razvoju web dizajna doprinjeo je dr. Jakob Nilsen, koji je razvio osnovne principe web mjesta (engl. Site), a vrlo jednostavnim za korištenje. Povijest web dizajna se može podijeliti na četiri generacije. Prvi period je počeo s razvojem prvog besplatnog pretraživača *Mozaik* a završio širokom rasprostranjenošću HTML-a. Dizajn web stranica u toj fazi je bio pod utjecajem mnogih tehnoloških ograničenja, kao što su spore modemske veze i nesposobnost pružatelja usluga za brz prijenos podataka. Za razliku od prvog perioda, web dizajn u drugom periodu izgleda više profesionalno i ima nove karakteristike, poput strukturiranog izbornika koji predstavlja hijerarhiju informacija. Javila se potreba za prilagodbom dizajn stranice određenim rezolucijama ekrana. Već su tada web dizajneri prepoznali kao značajan faktor brzinu preuzimanja informacija i rezoluciju monitora. Ipak tekst nije bio lak za čitanje zbog karakteristika pretraživača i brzine interneta, a korisnici nisu bili zainteresirani čekati nekoliko minuta za učitavanje sadržaja. Treća generacija web stranica - web dizajna kao glavni cilj imala je prikaz multimedijalnog sadržaja, uključujući zvuk, animaciju, 3D modele. Ova faza je povezana s uvođenjem *Flash* tehnologije koja je postala veoma popularna. Web dizajneri koristili su multimedijalni sadržaj kako bi privukli posjetitelje.

Dakle, struktura stranice i njegov navigacijski sistem smatrali su se izuzetno važnim za dizajn stranice, da bi se korisnicima omogućilo brzo pronalaženje onoga što traže. Osnovni princip u trećoj fazi bila je strategija privlačenja novih posjetitelja i zadržavanje istih na stranici što duže. Specifičnosti četvrte generacije su obilje multimedijalnih sadržaja uz mogućnost slanja preko interneta. Danas su stranice posebno razvijene za *ecommerce* (platforme za online prodaju), vlade, obrazovanje, zabavu i sl. [14, 17]

2.3. RESPONZIVNO WEB MJESTO

Web stranice koje će se morati razviti daleko su više izazovne nego što su bile. Potrebe rastu iz razvoja novih tehnologija te se zahtjevi povećavaju. Tri ključne tehničke značajke su temelj responzivnog web dizajna:

1. Upiti za medije;
2. Fleksibilan raspored na mreži koji koristi relativnu veličinu;
3. Fleksibilne slike i mediji, putem promjena CSS atributa.

Uspješan web dizajn temeljen na responzivnim načelima zahtijeva implementaciju svih navedenih značajki. Ključna stvar je prilagodba potrebama korisnika i mogućnostima uređaja. Ako se pretpostavi da će mobilni korisnik pregledavati web-lokaciju na malom zaslonu, potrebno je uzeti u obzir potrebe korisnika, što ne znači samo prilagodbu sadržaja na veličinu zaslona već i razmišljanje o tome što će mobilni korisnik prvo tražiti prilikom posjeta web-lokaciji, a zatim izraditi sadržaj u skladu s tim. Možda će biti potrebno mijenjati fontove ili područja interakcije. Dok mobilni uređaji mijenjaju pejzažni prikaz, uz pojavu sve više malih zaslona, ne smije se potisnuti što se događa na drugom kraju spektra. [3]

2.4. FLUIDNA MREŽA

Iza responzivnog dizajna važna je uporaba fluidne mreže (engl. Fluid Grid). Nedavno je stvaranje „fluidnog izgleda“, koji se širi sa stranicom, bio manje popularan od stvaranja izgleda fiksne širine, odnosno stranica koje imaju fiksno definiranu širinu *layout*-a uz fiksni broj piksela, a centrirane na stranici. Međutim, zbog ogromnog broja zaslonskih rezolucija prisutnih na današnjem tržištu, prednost *layout*-a prevelika je da bi se zanemarila.
Fluidne mreže nemaju tradicionalan izgled. Umjesto izrade izgleda na temelju piksela ili

proizvoljnih postotnih vrijednosti, fluidna mreža je pažljivije oblikovana u odnosu na proporcije. Kada je izgled stisnut na maleni mobilni uređaj ili se proteže preko velikog zaslona, svi elementi u izgledu mijenjaju širinu u odnosu jedni na druge. Ona predstavlja vrlo važan dio stvaranja responzivnog dizajna. Složeni izgled u tri stupca neće dobro funkcionirati na malom mobilnom telefonu, no, odgovarajući dizajn riješio je taj problem pomoću medijskih upita. Drugi dio responzivnog dizajna je CSS3 medijski upit i smatra se zadovoljavajućim u preglednicima. CSS medijski upiti u osnovi omogućuju prikupljanje podataka o posjetitelju web-lokacije i vraćaju CSS stilove za određenu dimenziju preglednika. U svrhe izrade stranice, značajna je medijska širina koja omogućuje primjenu određenih CSS stilova. Ako bi se primjenjivao neki stil za mobilne telefone, taj medijski upit bi izgledao slično kao u izvornom kodu broj 3. [4]

```
@media screen and (min-width: 480px) {
        .content {
                float: left;
                }
        .social_icons {
                display: none
                }
        // i tako dalje. . .
}
```

Izvorni kod broj 3. Medijski upit

Odgovarajući dizajn web stranica, trebao bi imati najmanje tri rasporeda za različite širine preglednika. Mali su širine ispod 600px, na taj način sadržaj će se prilagoditi većini telefona. Sredinu čine oni širine između 600px i 900px. Tako će sadržaj biti prilagođen većini tableta, nekim velikim telefonima, te određenim tipovima notebooka. Veliki su oni, čija je širina više od 900 piksela i tako će sadržaj biti prilagođen kao na većini osobnih računala. Svaki od tih izgleda trebao bi sadržavati isti tekst i grafičke elemente, ali svaki bi trebao biti dizajniran tako da najbolje prikaže taj isti sadržaj na temelju korisničkog uređaja. Smanjivanje stranice mora odgovarati manjim veličinama zaslona jer će inače učiniti sadržaj nečitljivim. Responzivni dizajn mora biti više od pretvaranja desktop stranice u verziju za mobilne zaslone. Potrebno je razmotriti korisničko sučelje, interakciju i sadržaj koji se traži dok se upotrebljava mobilni uređaj. [1]

3. CMS SUSTAV

CMS sustav je sustav za upravljanje sadržajem koji omogućava korisnicima da se bave isključivo sadržajem stranica, a ne detaljima programske izvedbe. U osnovi, takvi sustavi sadržaje web stranice drže u bazi podataka, a programi koji čine CMS, na zahtjev korisnika, od pohranjenih sadržaja dinamički stvaraju web stranicu. CMS sustav nadograđuje klasične web stranice ili drugim imenom, „statične“ web stranice. Statičnim se nazivaju jer se informacija nalazi upisana u HTML datotekama, često zajedno s programskim *kodom* koji ih prikazuje. Samo programer može izmijeniti ili obnoviti sadržaj postavljen na stranice te je potrebno dobro poznavati programske jezike u kojima su web stranice izrađene kako bi se sadržaj izmijenio. Obično su kodirane u HTML ili XHTML jezicima. Jezik HTML je originalno razvijen za kodiranje web stranica, a XHTML je njegova inačica sa striktnijom sintaksom. Neke od prednosti klasičnih web stranica su velika grafička fleksibilnost te mogućnost izrade kompleksne strukture. Glavni nedostatak je nemogućnost da sadržaj mijenja više ovlaštenih korisnika, bez programiranja i poznavanja detalja arhitekture web stranica. CMS je računalni program koji se koristi za stvaranje, izmjenu, upravljanje i objavljivanje sadržaja. Upravljanje sadržajem može uključivati slikovne medije, audio i video datoteke, elektroničke dokumente i ostali web sadržaj. Govoreći o prednostima, važno je spomenuti stabilnost, sigurnost, stalnu korisničku podršku te mogućnost nadogradnje u postojeće sustave i korisnikove baze podataka. Besplatni CMS sustavi su Joomla, Drupal, phpNuke, Typo3, Mambo, WordPress. Prednosti su im dostupnost i cijena.[2, 9, 12]

3.1.PROGRAMSKI JEZICI

Programski jezici omogućavaju realizaciju algoritama koji se mogu izvršavati na računalu. Interpretirani *kod* ne ovisi o arhitekturi računala na kojem se nalazi. Najpopularniji *server-side* web programski jezici su PHP, Perl i Phyton. CMS kreira stranice koje su u nekom od navedenih jezika. PHP (eng. Personal Home Page) koristi se za izradu dinamičkih web stranica. Koristeći CMS autor ne treba poznavati programski jezik, ali mora računalo opskrbiti alatima za izvođenje potrebnih radnji.

3.2.BAZE PODATAKA

Baze podataka pružaju mogućnost pohranjivanja podataka nad kojima je potrebno imati kontrolu i preglednost. U CMS-u svi podaci o korisnicima, lozinke, dozvole, sadržaji i sl., pohranjeni u bazama. MySql je sustav za upravljanje relacijskim bazama podataka, ali ne uključuje grafičko sučelje (GUI). Korisnici se mogu služiti s komandnom linijom (CLI) ako imaju znanje ili dodatnim programom koji će im pružiti grafičko sučelje. *phpMyAdmin* je alat pisan u PHP programskom jeziku. Služi za administriranje baza podataka i izvršavanje SQL naredbi kroz grafičko sučelje. [13]

4. WEB UI KOMPONENTE

Dizajn korisničkog sučelja (eng. User Interface - UI) fokusira se na osiguravanje da sučelje ima elemente koji su jednostavni za pristup, razumijevanje i upotrebu kako bi se olakšale te iste radnje. UI komponente povezuju pojmove iz interakcijskog dizajna, vizualnog dizajna i arhitekture informacija. Korisnici moraju biti upoznati s elementima sučelja koji djeluju na određeni način, moraju biti dosljedni i predvidljivi u samim izborima i izgledu sučelja. Korisnicima se tako povećava zadovoljstvo, učinkovitost te uspješnost u završetku zadataka. Elementi sučelja uključuju, ali nisu ograničeni na kontrole ulaza koje čine gumbi, polja s tekstom, potvrdne okvire, gumbe za odabir, okvire za popise, polje za datum i sl. Čine ih i navigacijske komponente kao što su polje za pretraživanje, brojanje stranica, klizač, oznake, ikone, informativne komponente od koji su najznačajnije opis alata, ikone, traka napretka, obavijesti, kutija za poruke, modalni prozori i dr. Prilikom oblikovanja sučelja svakako je potrebno znati da su najbolja sučelja gotovo nevidljiva korisniku i ne sadrže nepotrebne elemente. Potrebna je dosljednost i upotreba uobičajenih element korisničkog sučelja. Koristeći zajedničke elemente u korisničkom sučelju, korisnici mogu brže obavljati stvari. Također je važno izraditi obrasce jezika, izgleda i dizajna na cijelom web mjestu. Potrebno je razmisliti o prostornim odnosima između stavki na stranici te pažljivo postavljanje predmeta koji mogu pomoći privući pažnju na najvažnije informacije i pomoći u skeniranju i čitljivosti. Potrebno je strateški koristiti boju i teksturu. Pozornost može biti usmjerena prema ili preusmjerena s predmeta koji koriste boju, svjetlost, kontrast i teksturu u korist korisnika. Potrebno je koristiti tipografiju da bi se stvorila hijerarhija i jasnoća.

Važno je i upotrijebiti određenu vrstu pisma, različite veličine, fontove, raspored teksta kako bi se povećala mogućnost skeniranja i čitljivosti. Uvijek je potrebno obavještavati korisnike o lokaciji, akcijama, izmjenama ili pogreškama. U informacijskoj tehnologiji UI je dizajniran u informacijski uređaj s kojim osoba može komunicirati. To uključuje zaslone, tipkovnice, miševe i izgled radne površine. To je način na koji korisnik dolazi u interakciju s aplikacijom ili web stranicom. [13, 15, 16]

Prilikom početka rada na novom sučelju potrebno se usredotočiti na korisnika da bi se moglo stvoriti sučelje koje će im omogućiti da postignu svoje ciljeve. Potrebna je i dosljednost, jezik, izgled i dizajn. UI uvijek bi upozoriti korisnika kada su njegove aktivnosti ispravne, pogrešne ili pogrešno shvaćene. Važno je da se korisnici obavještavaju o radnjama, promjenama, pogreškama ili iznimkama koje se mogu dogoditi. Vizualni znakovi ili jednostavna poruka mogu pokazati korisniku da li su ga njegove radnje dovele do očekivanog rezultata. Potrebno je dizajnirati UI tako da korisnici lako mogu poništiti radnje i razne unose. [14, 15]

5. FRONTEND I BACKEND TEHNOLOGIJE U KONTEKSTU WEB DIZAJNA

5.1. FRONTEND TEME

Korisničko sučelje je dio weba koji se može vidjeti i s kojim se može komunicirati. UI se obično sastoji od web dizajna i *frontend* web-a. Razvoj je krenuo dalje kada su dizajneri počeli sa JavaScript-om i jQuery-em. Sve što je vidljivo pri korištenju web-a, kombinacija je HTML-a, CSS-a i JavaScript-a i sl. , a kontrolirano je preglednikom računala. Razvojem *frontend* dijela preglednika računala, Techopedia definira kao: „Razvoj *koda* koji stvara vizualne elemente *frontend* dijela softvera, aplikacije ili web mjesta“[4]. Rad *frontend* softvera uglavnom je usredotočen na angažiranje u analizi *koda*, dizajna i programskih pogrešaka, upravljanje onim što ljudi prvi put vide u pregledniku. *Frontend* programer ima odgovornost za izgled i dizajn web mjesta. [7, 10]

[4] Techopedia: „Definition – What does Front-End Developer mean?“
URL: https://www.techopedia.com/definition/29569/front-end-developer

5.2. JEZICI ZA KODIRANJE FRONTEND-a

Frontend čine opisni jezici HTML i CSS i programski jezik JavaScript. Korištenje jQuery-a, također se često povezuje s razvojnim radom na *frontend*-u. To su samo neki od primjera programskih jezika. Prilikom izrade projekta spominje se i responzivan dizajn uz tipografiju, *font*-ove, *grid* i teoriju boja, a značajno je stvaranje i redizajniranje web-lokacija. Važno je da je sadržaj fiksiran, tj. da se veliki dijelovi novih podataka neće stalno prenositi. [7]

5.3. PREDUVJETI ZA KODIRANJE FRONTEND-a

Realizacijom *frontend* dijela web mjesta bave se UI i UX dizajneri. Njihov zajednički naziv je web dizajner. UI (engl. user interface) i UX (engl. user experience) dizajneri su *frontend* programeri koji se usredotočuju na korisničko sučelje. UI dizajneri bave se vizualnim aspektima dizajna web-lokacije, a UX dizajneri provode testiranja korisnika kako bi se osiguralo da web-lokacija dobro funkcionira s korisnicima. Izazov povezan s razvojem *frontenda* je to što se alati i tehnike korištene za stvaranje *frontend-a* web stranice neprestano mijenjaju. Cilj izrade web mjesta je osigurati korisnicima vidljivost informacija u obliku koji je jednostavan za čitanje te relevantnost informacija. Potrebno je osigurati da se njihova web lokacija pravilno pojavljuje u različitim preglednicima (engl. Cross - Browser), različitim operacijskim sustavima (engl. Cross - Platform) i različitim uređajima (engl. Cross - Device), što zahtijeva pažljivo planiranje od strane programera. Postoje dostupni alati koji se mogu koristiti za razvoj *frontend-a* web stranice ali je uz njih potrebno i razumijevanje alata kako bi se odabrao najbolji za određene zadatke. [14, 15, 16]

5.4. BACKEND PANEL

Backend panel je administratorska ploča koja se definira kao mjesto gdje se stvaraju novi *post*-ovi, kategorije, oznake, stranice i veze. Također, ona je mjesto gdje se kreira sadržaj i upravlja web stranicom. Ključ je za funkcioniranje sustava za upravljanje sadržajem i osnovna ideja ploče s instrumentima koja daje mjesto pregleda svega što se događa s napravljenim *blog*-om.

Područje *backend panela* daje opći pregled web stranice, također prikazuje mnoge korisne brze veze za obavljanje uobičajenih zadataka poput pisanja brzih skica ili odgovaranja na najnoviji komentar. Na njoj se mijenjaju tematske datoteke, dodaju *widgeti*, aktiviraju ili ažuriraju podaci, mijenjaju opće postavke čitanja i pisanja. *Backend panel* je samo još jedan od niza oblika koji omogućuju sve gore navedeno, bez ručnog uređivanja i prijenosa datoteka. Posao bez *backend panela* bio bi znatno otežan. Kako WordPress omogućuje minimizaciju *widgeta*, također omogućuje povlačenje *widgeta* na novu lokaciju. To omogućuje postavljane *widgeta* koji se često koristite na istaknutiji položaj. Za mnoge ljude, *backend panel* jednostavno je stranica koju vide prije nego što pritisnu na izbornik administratora i prijeđu na određenu stranicu. Za ostale, *backend panel* je postala „neugodna“ zbog nepotrebnih beskorisnih *widget*-a iz programskih dodataka. WordPress je taj koji daje alate za upravljanje, a sam korisnik ih mora znati koristiti. Zaslon za administraciju omogućuje pristup kontrolnim značajkama instalacije programa WordPress. Svaki zaslon administracije prikazan je u odjeljcima, alatnoj traci (na zaglavlju), glavnoj navigaciji, radnom području i u podnožju. Mnoge stavke alatne trake proširuju se za prikaz više informacija. S lijeve strane zaslona glavni je navigacijski izbornik s pojedinostima o administrativnim funkcijama koje se mogu izvoditi. Na dnu tog odjeljka je gumb za sažimanje izbornika koji smanjuje izbornik u skupinu ikona ili se širi kako bi ih se popisala glavnom funkcijom. Veliko područje na sredini zaslona je radno područje. Na njemu su prikazane i prikupljene konkretne informacije vezane uz određeni navigacijski izbor, kao što je dodavanje novog *posta*. [9, 12]

Konačno, u podnožju, pri dnu svakog zaslona administratora veze su na WordPress te je prikazana verzija programa WordPress-a koja je instalirana. Kao i u *frontend*-u, postoje tri najvažnija programska jezika, koja nazivamo standardiziranima. HTML za označavanje, CSS za prezentaciju te JavaScript za skriptiranje. Svaki preglednik brine o tome da li su pravilno oblikovane HTML, CSS i JavaScript datoteke. Pozadina se obično sastoji od tri dijela, a to su poslužitelj, aplikacije i baze podataka. Nakon unesenih podataka, aplikacija ih pohranjuje u bazu podataka koja je napravljena na poslužitelju. Sve te informacije ostaju na poslužitelju tako da se prilikom ponovne prijave u aplikaciju sve informacije još uvijek nalaze na istome mjestu. [12]

Backend tehnologije obično se sastoje od jezika kao što su PHP, Ruby, Python i sl. Radi lakše upotrebe, obično ih poboljšavaju radni okviri poput Ruby on Rails, Cake PHP i Code Igniter koji čine razvoj bržim i jednostavnijim za uporabu.

WordPress je dobar primjer *frontenda* i *backenda* koji rade zajedno, jer je WordPress open-source okvir izgrađen na PHP-u koji je potrebno instalirati na poslužitelju s bazom podataka, a zatim su tu dizajneri koji prilagode izgled i funkcionalnost WordPress web stranice koristeći CSS, jQuery i JavaScript. [9, 12]

Backend pokreće web lokaciju, dok ga korisnik ne vidi ili ne može izravno komunicirati s njim, ali uvijek se pokreće u pozadini, pružajući glatku funkcionalnost i iskustvo na računalima i šalje informacije iz baze podataka izravno u preglednik. *Backend kod* daje funkcionalnost svemu što stvara, kombinacija je baze podataka i softvera pisanog na strani poslužitelja, a koji se izvode na web poslužiteljima (engl. Server). Ova aplikacija na strani poslužitelja izravno komunicira s bazom podataka putem sučelja programa (API), koji povlači, sprema ili mijenja podatke (CRUD). Podaci se vraćaju i pretvaraju u *frontend kod* s kojim korisnik ostvaruje interakciju s popunjavanjem obrasca, stvaranjem profila, online kupnjom itd. Općenito, sve što se vidi na nekom web mjestu omogućeno je *backend kodom* koji je smješten na poslužitelju i koristi ga. *Backend* programeri stvaraju i održavaju cjelokupnu gore navedenu funkciju. [12]

5.5. CRUD FUNKCIONALNOSTI

Create, *read*, *update* i *delete* su temeljne funkcije svake dinamične aplikacije. To je forma za pregled podataka uz mogućnost dodavanja, uređivanja i brisanja podataka. CRUD operacije su ono s čime se svaki web developer susreće. Želeći napraviti neki projekt, potrebno je u web pregledniku otvoriti adresu, odabrati formu za unos, pregled, uređivanje i brisanje sadržaja te dodati podršku za Bootstrap kako bi se komponente Bootstrap-a mogle koristiti. Potrebno je kreirati servis koji se koristi za različite svrhe, a na njemu je moguće vidjeti sve od pregleda svih korisnika, dodavanja novih, uređivanja i brisanja postojećih. Nakon definiranog izgleda forme, kreiraju se opcije koje će joj dati funkcionalnost. Mogu se dodavati korisnici. Kao zadnji korak možemo navesti brisanje (eng. Delete). Isto se radi na primjerima s WordPressom uz napomenu da se koriste različiti programski jezici. [17]

6. BOOTSTRAP RADNI OKVIR I NJEGOVE KOMPONENTE

Bootstrap je set HTML, CSS i JS komponenti koje su namijenjene brzoj izradi web stranica. On je *frontend* radni okvir (eng. Framework) koji sadrži besplatnu kolekciju alata za stvaranje web-stranica i web-aplikacija. Uključuje HTML i CSS predloške za gumbe, navigaciju, obrasce te JavaScript proširenja. Početnu verziju izradili su Mark Otto i Jacob Thorton 2010. godine, dok su radili kao dizajneri u tvrtki Twitter Inc. Alat je inicijalno namijenjen očuvanju dosljednosti u dizajnu korisničkih sučelja, no s vremenom je postao slobodno dostupan korisnicima kao *open source* softver. Jednostavnom modifikacijom CSS *koda* moguće je brzo izmijeniti grafiku HTML element, zbog čega je Bootstrap postao najpopularniji radni okvir za web dizajn. [1]

Frontend developeri samostalno pišu svoj HTML, CSS i JavaScript *kod* i šalju teme *backend* developerima kako bi oni spojili statičnu stranicu na CMS sustav, pomoću PHP-a na bazu podataka i tako ih napravili dinamičnima. *Backend* developer radi napredni posao jer stavlja stranicu u funkciju. Bootstrap se koristi za lakšu izradu izgleda stranice. *Frontend* developeri pišu svojstveni *kod* za svaku stranicu a time se sa svakom novom stranicom piše isti *kod* za elemente ispočetka što oduzima puno vremena. Bootstrap-om možemo uštedjeti ogromnu količinu vremena jer već ima ugrađene elemente koji se ne trebaju pisati iznova za svaku stranicu, kao *responsive grid*. Kod izrade vlastite stranice, najbolje je koristiti Bootstrap radi uštede vremena, radi same forme, stiliziranja tipografije, ikone, upozorenja, navigacije, tabele i sl. U objavljenom blogu, Mark Otto je ovako predstavio projekt: „U ranijim danima Twittera, inženjeri su koristili gotovo svu knjižnicu (eng. Library) s kojom su upoznati kako bi zadovoljili preduvjete. Nedosljednosti između pojedinačnih izmjena teško je mjeriti i održavati. Bootstrap je počeo kao odgovor na upravo te izazove i ubrzo se počeo povećavati tijekom prvog Hackweeka na Twitteru. Do kraja Hackweeka imali smo stabilnu verziju koju bi inženjeri mogli koristiti diljem tvrtke. “[5] Početak razvoja obilježen je projektom koji je u potpunosti upravljan CSS-om te uključivanjem niza JavaScript dodataka i ikona koje idu razmjerno s formama i gumbima. U svojoj bazi on podržava responzivni Web dizajn i ima 940 piksela široku mrežu. Na Bootstrap web stranici se može izabrati izgled i dodatci koji su potrebni na stranici, a to su CSS i JavaScript dodaci koji ubrzavaju izradu web stranice. [1, 4]

[5] Spurlock, Jake: „Bootstrap“, O'Rilley, SAD, 2013.

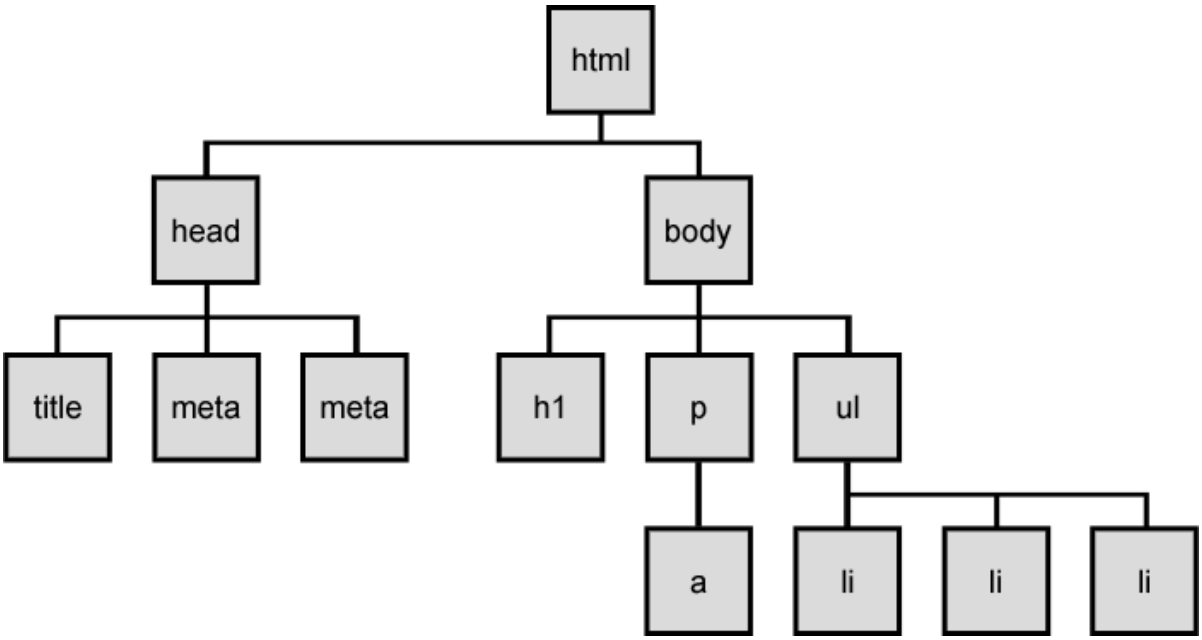
6.1. HTML

HTML je opisni jezik za izradu web stranica. Izvršava se u web pregledniku te definira strukturu i sadržaj stranice. On je sve što se vidi na stranici i što piše. Da bi se napravila ispravna HTML datoteka, unutar dokumenta potrebno je napisati tri oznake: <*HTML*>, <*head*> i <*body*> te posebni prvi red sa sljedećom oznakom <*!DOCTYPE HTML*>, koju ne treba zatvarati. HTML oznakom se uokviruje cijeli dokument, kako bi web preglednik znao da se radi o HTML *kodu*. Unutar njega mora biti element <*head*> i <*body*>. U <*head*> element se stavljaju opće informacije o stranici, kao što su naslov (eng. Title) te veze s drugim dokumentima koje želimo očitati prije otvaranja stranice. Taj dio stranice se ne prikazuje u web pregledniku. <*Body*> označava „tijelo“ stranice. U taj element stavljamo sve elemente koje želimo prikazati kao što su paragrafi i slike. Prilikom spremanja umjesto uobičajene *. *txt* ekstenzije, datoteku spremimo sa *. *html* ekstenzijom i otvorimo je u web pregledniku kako bi vidjeli što smo postigli. U izvornom *kodu* broj 4 je prikazana najjednostavnija web stranica. [13]

```
<!DOCTYPE HTML>
<HTML>
	<head>
		<title>Page Title</title>
	</head>
	<body>
		<h1>This is a Heading</h1>
		<p>This is a paragraph.</p>
	</body>
</HTML>
```

Izvorni kod broj 4. Struktura HTML dokumenta

Osnovni dio web stranice čine HTML elementi. Internet preglednici interpretiraju HTML elemente da bi iscrtali web stranicu, a oni uobičajeno imaju početak i kraj (npr. <h1> . . . </h1>), mogu imati i prazne elemente (npr.
), mogu sadržavati atribute (npr. <h1 id=“remember-it”> . . . </h1>), mogu se umetati jedan unutar drugoga te zajedno čine hijerarhiju (Slika broj 1). Preporučuje se da HTML elementi budu zatvoreni, a atributi unutar navodnika. [13]



Slika 1. Elementi HTML dokumenta, izvor: codepen: „Online - path selector“[23]

6.2. CSS

CSS je (eng. Cascading Style Sheets) stilski jezik, koji se koristi za opis dizajna dokumenta napisanog pomoću HTML jezika. Izvršava se u web pregledniku. Datoteka ima ekstenziju **.css*. Dakle, CSS definira izgled stranice. Primjer CSS deklaracije čine: p – selektor, s time da se deklaracija odnosi na sve <*p*> elemente te unutar zagrada slijede pravila, boja (eng. Color) kao svojstvo, crvena (eng. red) kao vrijednost te CSS pravilo koje završava znakom točka-zarez (;)(Izvorni *kod* broj 5). [13]

```
p {
    color: red;
    text-align: center;
}
```

Izvorni kod broj 5. CSS deklaracija

Kako se HTML razvijao, počeli su se koristiti elementi koji definiraju *font*, veličinu teksta, boju teksta, širinu tablica i druge elemente za stiliziranje stranice, iako je HTML namijenjen sadržaju. Tu se javlja problem jer se u HTML datotekama teže čita i mijenja sadržaj zbog velike količine stilskih naredbi.

Problem se riješio pojavom CSS-a, koji se koristi unutar HTML datoteke sa oznakom <*style*> ili u zasebnoj datoteci jer se na taj način definira stil za sve iste elemente na stranici. CSS pravila se primjenjuju na elementima koji su odabrani selektorima. Osnovne vrste selektora su: 1. odabir elementa; 2. odabir prema identifikatoru elementa; 3. atribut naziva *id* s time da dva elementa na web stranici ne smiju imati istu vrijednost atributa *id*; 4. odabir prema klasi elementa; 5. atribut naziva *class* od kojih više element može imati istu klasu; te 6. kombinacija svega navedenog. Elementi nasljeđuju pravila od nadređenog HTML elementa. [13]

6.3. JAVASCRIPT

JavaScript je skriptni programski jezik koji se izvršava u web pregledniku. Datoteka ima ekstenziju **.js* Za razliku od CSS-a i HTML-a, JavaScript je programski jezik koji dodaje web stranicama interaktivnost i dinamičnost. S njim možemo deklarirati varijable, zbrajati, množiti, raditi funkcije, nizove, komparacije i sve ostale mogućnosti. On se izvršava u pregledniku te se koristi i za serverske aplikacije – *Node.js*. Dinamički mijenja HTML, CSS čime se mijenja struktura i izgled stranice. Reagira na korisničke akcije mišem, tipkovnicom, dodirom, a što označuje događajima (eng. Events). Nadalje, komunicira sa serverom bez ponovnog učitavanja cijele stranice – *AJAX*. Najčešće se koristi u obliku funkcije koja se aktivira pomoću nekog događaja u HTML djelu stranice, npr, na stisak gumba. Za korištenje JavaScript-a, *kod* se mora ubaciti u HTML datoteku unutar elementa <*script*> ili posebnom naredbom ako je želimo pozvati iz zasebne datoteke (slično kao i za CSS datoteku). Početni JavaScript prikazan je u izvornom *kodu* broj 6. [13]

```
<!DOCTYPE HTML>
<HTML>
<body>
	<h2>My First JavaScript</h2>
<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click me to display Date and Time.</button>
	<p id="demo"></p>
</body>
</HTML>
```

Izvorni kod broj 6. Početni JavaScript kod

Uključivanje JavaScript-a na stranici počinje korištenjem atributa što se jasno vidi iz početnog JavaScript primjera navedenog gore u tekstu. Nadalje, uključivanje uključuje i korištenje sadržaja <*script*> elementa iz vanjske JS datoteke. JavaScript ne reagira na neke greške što nije osobito dobra osobina programskog jezika. Obično su to one greške koje je teško uočiti i ispraviti. [13]

6.4. BOOTSTRAP MAPA

```
Css/   Bootstrap.css
       Bootstrap.css.map
       Bootstrap.min.css
       Bootstrap.min.css.map
       Bootstrap-theme.css
       Bootstrap-theme.css.map
       Bootstrap-theme.min.css
       Bootstrap-theme.min.css.map
fonts/ glyphicons-halflings-regular.eot
       glyphicons-halflings-regular.svg
       glyphicons-halflings-regular.ttf
       glyphicons-halflings-regular.woff
       glyphicons-halflings-regular.woff2
js/    Bootstrap.js
       Bootstrap.min.js
       npm.js
```

Izvorni kod broj 7. Bootstrap mapa

Kako je navedeno, Bootstrap ima 3 mape: *css*, *fonts* i *js*, a u njima postoje minimizirane i regularne verzije CSS-a i JavaScript datoteka, potrebne za izradu web stranice, no nije nužno imati obje verzije. U praksi se, za izradu web stranica, najviše koriste regularne datoteke, jer postoji mogućnost promjene u samom *kodu* CSS-a i zbog same preglednosti, a kada se stranica stavlja na server koriste se minimizirane verzije Bootstrap CSS-a. Bootstrap ima ogromnu količinu *koda* i dodataka koje olakšavaju i ubrzavaju izradu web stranice. [11]

U Bootstrap je uključeno mnoštvo zasebnih programskih ekstenzija, poput *normalize.css* koji normalizira renderiranje stranice za pregled u različitim web preglednicima, postavlja pozadinsku stranicu na *body*-u u bijelu (engl. background-color: #fff;), koristi *@font-family-bas*, *@font-size-base* i *@line-height-base* kao osnovu za fontove. Na stranici i svim linkovima postavlja boju sa *@link-color* te dodaje crtu ispod imena linka na :*hover*. Ti se stilovi nalaze pod *scaffolding.less*. [1, 11, 13]

6.5. BOOTSTRAP TIPOGRAFIJA

Bootstrap sadrži naslov (eng. Heading) te *heading* elemente počevši sa <*h1*> veličinom fonta od 36px, pa sve do <*h6*> s veličinom od 12px, dok je standardna veličina teksta 14px. Bootstrap sadrži nekoliko predefiniranih klasa i elemenata koji se mogu koristiti za stiliziranje teksta. Primjerice, <*small*> ili *small* klasa koje smanjuju tekst i posvjetljuju ga. Klasa *strong* čini tekst podebljanim, za dobivanje *kurziva* tekst se stavlja unutar <*em*> elemenat (engl. emphasis) s ciljem naglašavanja pojedinih dijelova teksta. Uz navedene, Bootstrap ima još neke klase za naglašavanje teksta kao, *muted*, *text-warning*, *text-error*, *text-info*, *text success* kao u izvornom *kodu* broj 8. [1, 11]

```
<p class="text-warning">Ovaj tekst će biti naglašen</p>
```

Izvorni kod broj 8. Bootstrap klasa za naglašavanje

Za dodavanje citiranog teksta postoji naredba *blackquote*, s kojom uz pomoć <*p*> element dobivamo tekst koji je uvučen s lijeve strane. [1, 11]

6.6.LISTE I TABELE

Bootstrap uz klasične liste, posjeduje liste s redoslijedom i bez redoslijeda, te nudi liste s definicijama. Klasične liste se ponašaju kao i u običnom HTML-u dok definirane liste umjesto standardnih <*li*> elemenata imaju i sljedeće elemente: <*dt*> (engl. definition term što u prijevodu znači pojam definicije) i <*dd*> element koji označava definiciju <*dt*> elementa. Bootstrap ima već ugrađene stilove za izgled tablica, tako da se dobiva uredan dizajn tablica. Za tablicu u kojoj su redovi odvojeni tankom linijom koristi se *table* klasa i to je osnovna

klasa za stiliziranje tabele. Uz *table* postoji još i klasa kao *table-striped* koja svakom drugom redu u tablici daje pozadinsku boju. *S table-bordered* svaki element dobiva granice unutar tablice, obrubljuje ih. Da bi se dobio *hover efekt* na redovima tablice dodaje se klasa *table-hover* koja prilikom prelaska kursora preko pojedinog reda ističe taj red. U slučaju da je tablica veća od željene veličine, s *table-condensed* klasom smanjuju se *padding* redova na pola što i smanjuje cijelu tablicu. Redovi u tablici se također mogu stilizirati s klasama koje mijenjaju pozadinske boje redova. Za dobivanje responzivne tablice dodaje se klasa *table-responsive*. [1, 11, 13]

6.7.FORME

Bootstrap sadržava predefinirane stilove za tri različita stila formi, a to su vertikalne ili klasične forme, *inline* forme i horizontalne forme. Kod vertikalnih formi ne dodaju se dodatne klase, što se vidi u izvornom *kodu* broj 9.

```
<form action="/action_page.php">
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control" id="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control" id="pwd">
  </div>
  <div class="checkbox">
    <label><input type="checkbox"> Remember me</label>
  </div>
  <button type="submit" class="btn btn-default">Submit</button>
</form>
```

Izvorni kod broj 9. Bootstrap forma

Za *inline* formu samo u klasi forme dodamo *form-inline* te će tako elementi stajati jedan iza drugoga, a ne jedan ispod drugoga.

Za dobivanje horizontalne forme, u <*form*> elementu navodi se klasa *form-horizontal*. Unutar forme imena (engl. Label) i polja, se odvajaju <*div*> elementima s klasom *form-group* i poljima se dodaju klase *control-label*, kao u izvornom *kodu* broj 10. [11, 13]

```
<form class="form-horizontal" action="/action_page.php">
  <div class="form-group">
    <label class="control-label col-sm-2" for="email">Email:</label>
    <div class="col-sm-10">
      <input type="email" class="form-control" id="email"
placeholder="Enter email">
    </div>
  </div>
  <div class="form-group">
    <label class="control-label col-sm-2"
for="pwd">Password:</label>
    <div class="col-sm-10">
      <input type="password" class="form-control" id="pwd"
placeholder="Enter password">
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <div class="checkbox">
        <label><input type="checkbox"> Remember me</label>
      </div>
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <button type="submit" class="btn btn-default">Submit</button>
    </div>
  </div>
</form>
```

Izvorni kod broj 10. Bootstrap horizontalna forma

Bootstrap ima podršku za sve HTML unose, tekstove, zaporke, vremena, datume, mjesece, tjedne, brojeve, Email-ove, URL, pretraživanja, telefone i boje. [1]

6.8. GUMBI I SLIKE

U Bootstrapu svim elementima kojima je dodijeljena klasa *btn* dobivaju standardni izgled sivog gumba sa zaobljenim rubovima. Također, definirani su izgledi gumbova, primjerice, s klasom *btn-primary*, gumb dobiva plavu boju koja mu daje dodatnu vizualnu težinu i pravi ga primarnim u nizu gumbova. Svijetloplava boja dobiva se sa *btn-info*, zelena sa *btn-success*, crvena sa *btn-danger*, narančasta boja sa *btn-warning*. Gumbima se mogu mijenjati veličine s predefiniranim klasama Bootstrapa, *btn-large* ili skraćeno *btn-lg*, *btn-small* i *btn-xs* za jako male gumbe. Postoji i mogućnost dodavanja gumbova u blokove s klasom *btn-block*. Slike imaju tri klase koje se mogu koristiti u Bootstrapu. Za dodavanje radijusa na sliku, koristi se klasa *img-rounded* koja slici daje *border-radius* od 6px, sa *img-circle* cijelu sliku čini okruglom dajući joj *border-radius* od 500px i sa *img-polaroid* daje slici *padding* i sivu obod. Kako bi slike bile responzivne i lijepo se skalirale, Bootstrap podržava klasu *img-responsive* koja daje slici maksimalnu širinu od 100% (engl. max – width = 100%), automatsku visinu (engl. height – auto) i prikaz u blokovima na ekranu (engl. display – block). [1, 11, 13]

6.9. GLYPHICONS

Bootstrap sadržava 260 ikona pohranjenih u jedan *sprite*, a koje se mogu koristiti na gumbima, linkovima, navigacijama i formama. Kod korištenja ikona unutar *koda*, moraju se upisati klase za dohvat ikona. Prva klasa koja se upisuje je *glyphicon* koja govori pretraživaču da je riječ o ikoni, a druga klasa koja se upisuje je klasa koja određuje ikonu koju želimo staviti na određeno mjesto, kao u izvornom *kodu* broj 11.

```
<span class="glyphicon glyphicon-name"></span>
```

Izvorni kod broj 11. Bootstrap Glyphicon klasa

6.10. RAZLIKE IZMEĐU STATIČNIH I DINAMIČNIH WEB STRANICA

Statične web stranice su kao brošure, napravljene pomoću HTML-a i CSS-a. To su HTML stranice kojima je CSS napravio izgled, a HTML strukturu stranice. Tekstovi na stranici su pisani unutar *koda* i spremljen u HTML datoteku. Kod dinamičnih stranica informacije se dobivaju iz baze podataka, osvježavaju se od strane korisnika ili samih dizajnera stranice.

6.11. WIREFRAMING

Wireframing je važan korak u izradi stranice, shema web stranice, a u većini slučajeva se preskače. On daje mogućnost razrade svih detalja prije nego se napravi veći dio, pa tek onda dolazi do izmjena koje su se previdjele na početku. Lakše je modificirati *wireframe* nego cijele dijelove *koda*. To je kao skica, korisnik se više može posvetiti rasporedu elemenata, u formama, linkovima, navigacijama i sl. Kada postoje skice, u photoshopu je lakše i brže iscrtati stranice jer već postoji shema. [11]

6.12. SKIDANJE I INSTALACIJA BOOTSTRAPA

Najlakši način za preuzimanje i instalaciju Bootstrapa je preuzimanje sa *„getbootstrap. com“* web stranice. Na sredini stranice je gumb s nazivom *Download Bootstrap* za preuzimanje Bootstrap datoteka. Dok se stisne gumb *Download Bootstrap,* dobivaju se tri opcije: *Download Bootstrap*, *source code* i *sass* koji je CSS pretprocesor. *Source code* daje *Less,* JavaScript i *font* datoteke i potreban je *Less* pretprocesor za CSS. Kod opcije *Download* dobiva se *Bootstrap-x.x.x.zip* datoteka u kojoj se nalaze Bootstrap CSS datoteke, *fontovi* i JavaScript skripte. [11]

6.13. POSTAVLJANJE RADNE OKOLINE PROJEKTA I KODIRANJE

Za početak je potrebno otvoriti praznu mapu u koju se pohranjuje cijela web stranica. Unutar glavne mape kopiraju se datoteke i mape koje se nalaze u *Bootstrap.zip* datoteci. U CSS mapi izbrišu se nepotrebne datoteke osim *bootstrap-theme.min.css* i *bootstrap.min.css*.

Datoteke se brišu jer korisnik sam piše svoj CSS i ostavlja samo minimizirane verzije Bootstrap CSS-ova. U *tekst editoru* napravi se nova *index.html* datoteka. U HTML-u, unutar <*head*> djela, potrebno je napraviti <*meta*> element kako bi web preglednici imali više informacija o napravljenoj stranici, kao što je navedeno ispod.

```
<meta http-squiv=“Content-Type“ content=“text/HTML; charset=UTF-8“>
```

Izvorni kod broj 12. HTML content type

Sljedeći <*meta*> element govori uređajima početnu širinu i veličinu *viewporta*.

```
<meta name=“viewport“ content=“width=device-width, initial-scale=1“>
```

Izvorni kod broj 13. HTML viewport

Potreban je i prazan <*meta*> element koji se uređuje i mijenja tijekom izrade stranice.

```
<meta name="description" content="">
```

Izvorni kod broj 14. HTML description

Sljedeći <*meta*> element postoji jer što će biti dopunjen (engl. update) u WordPressu s nazivom autora stranice.

```
<meta name="author" content="">
```

Izvorni kod broj 15. HTML author

7. WORDPRESS CMS SUSTAV I NJEGOVE KOMPONENTE

WordPress je sustav za upravljanje sadržajem (engl. CMS). Sustavi za upravljanje sadržajem definiraju se kao proizvodi programske podrške koji omogućuju objavu, uređivanje i pohranu sadržaja. Njihova namjena i opseg, variraju od rješenja namijenjenih upravljanju dokumentima u velikim organizacijama do sustava koji služe za objavu korisničkih blogova. WordPress je trenutno globalno najčešće korišten sustav za upravljanje sadržajem, a razvijen je 2003. godine s ciljem olakšanja aktivnosti pisanja i objave sadržaja na webu. Primarno je bio namijenjen za objavu osobnih blogova, ali dodavanjem funkcionalnosti tijekom godina postao je cjeloviti sustav za upravljanje sadržajem. Glavne značajke sustava WordPress su jednostavnost te veliki izbor modula i priključaka koji omogućuju fleksibilnost i prilagodbu sustava specifičnim potrebama korisnika. [9, 17]

Projekt lokalizacije WordPressa na hrvatski jezik započeo je 2007. godine s inačicom WordPress 2.2.2. Prvi korak u lokalizaciji WordPressa na hrvatski jezik je bilo prevođenje lokalizacijske *.pot* datoteke te kreiranje lokaliziranih *hr.po* i *hr.mo* datoteka te njihova distribucija zainteresiranim korisnicima. U planu je također bilo i lokaliziranje ostatka instalacijskog WordPress paketa te njegovo objavljivanje kao cjelokupnog lokaliziranog instalacijskog paketa na hrvatskom jeziku. U isto vrijeme postavljena je i službena hrvatska WordPress stranica koja bi služila za objavljivanje novih lokaliziranih inačica WordPressa kao i svih novosti vezanih za hrvatsku lokalizaciju. Od inačice 2.2.2 pa sve do inačice 2.7 objavljivanje prevedenih lokalizacijskih datoteka obavljalo se na neslužbenoj stranici WordPress Croatia, nakon čega nastaje stanka u objavljivanju. Godine 2010. projekt lokalizacije WordPressa na hrvatski jezik se oživljava i ponovno pokreće. WordPress je besplatan, a može se preuzeti sa stranice *wordpress.org*, i instalirati. Besplatne su i sve nadogradnje, a dodaci (engl. Plugins) kojima se proširuje funkcionalnost također su većinom besplatni. Postoji mogućnost preuzimanja besplatnih tema, no većina tih tema namijenjena je blogerima. WordPress, pošto je open source projekt, dolazi s velikom mogućnošću proširenja. Postoji veliki broj dodataka kojima se može povećati funkcionalnost stranice te se tako može dodati anketa, galerija fotografija, video ili kontaktna formu i s time prilagoditi web stranica i WordPress vlastitim potrebama. Ono što dijeli WordPress od ostalih CMS sustava jest jednostavnost, kvalitetan dizajn, širok spektar aplikacija te odlična podrška od strane korisnika, klijenata i programera. Veoma je prilagodljiv sustav. „Dovoljno je reći podatak da 30% svih web stranica na cijelom Internetu su unutar WordPress sustava, naravno uz to je i

najpopularnija blogerska platforma s preko 60 milijuna web blogova“[6]. Karakteristike i prednosti dizajnerske mogućnosti u pogledu tema i nadogradnji su vrlo velike. S obzirom na to da je WordPress od početka bio blogerska platforma, takva forma je ostala i danas što se vidi pretežito na kontrolnoj ploči (engl. Dashboard). To se vidi nakon instalacije i prijave administratora gdje se najbitnije stvari nalaze s lijeve strane u meniju (*postovi*, stranice, komentari, multimedije odnosno repozitorij za prijenos multimedije, mogućnost dodataka novih nadogradnji, izgled same web stranice u koju se ubrajaju i teme, korisničke postavke, općenite postavke i ostalo). Ako se instalira neki dodatak putem *plugin storea*, njegova stavka se dodaje izborniku sa strane za lakše snalaženje i dodatne opcije. Sučelje i rad za samog korisnika je veoma jednostavno i jasno. Prije početka izrade same web stranice moramo odabrati određenu temu odnosno predložak koji će služiti kao dizajn naše stranice. Nakon odabira teme, ako smo zadovoljni sa samim izgledom teme možemo početi s upisom i uređivanjem sadržaja. U slučaju kada nismo zadovoljni s izgledom naše web stranice, tada dolazi na red instalacija *pluginova* tj. nadogradnji. [9, 17]

Trenutno postoji oko 40 tisuća raznih nadogradnji koji omogućuju potpuno individualiziranje izrađenih web stranica. Uz to postoji opcija manualnog uređivanja HTML i CSS *koda* što koriste iskusniji programeri. Ako se stvori profitabilnu ili poznata web stranica, WordPressova podrška za *SEO* optimizaciju je vrlo snažna, dok će razne tražilice dati prednost WordPress stranicama. [2, 9]

7.1. NADZORNA PLOČA

Nadzorna ploča središnje je mjesto WordPress CMS sustava. Administrator (osoba s najvećim pravima) ima pristup svim dijelovima koji većinom nisu dostupni ostalim korisnicima. Pomoću njih mogu se dodavati novi i uređivati već postojeći sadržaji, nadograđivati stranica s dodatcima, alatima i dr. Ovisno o tome što se želi dodati na web prostor, odabire se određena stavka iz izbornika. *Postovi* služe za izradu, objavu i izmjenu članaka, kategorija (engl. Categories) i *tagove*. Svaki novi *post* automatski se objavljuje i na naslovnoj stranici. Postavljanjem kategorije na članke postoji mogućnost da se izdvoje svi članci određene kategorije (na samoj stranici). Mediji se koriste za upravljanje medijskim datotekama (slike, dokumenti, glazba…) koje se mogu dodati u članke ili na stranice. Medijska zbirka (eng. Library) sadrži prikaz svih medijskih datoteka korištenih u sustavu.

6

Poveznice (eng. Links) koriste se za upravljanje poveznicama prema drugim web stranicama. Osim pregleda postojećih i dodavanja novih, mogu se i kategorizirati. Stranice (engl. Pages) su statični dijelovi, najčešće se koriste za sadržaje koji se rijetko mijenjaju (poput stranice kontakta). Stranice se mogu povezati s izbornikom, no na stranicama se ne objavljuju novi članci kao što je to u slučaju s naslovnom stranicom. Komentari (engl. Comments) služe za upravljanje komentarima. Ako su podešeni da se komentar najprije mora odobriti, također se to čini kroz isto sučelje.

Izgled (engl. Appearance) se koristi za upravljanje izgledom same stranice (temom koja se koristi). Kroz ovaj dio mogu se mijenjati teme, dodavati *widgeti*, uređivati izbornici, uređivati zaglavlja (engl. Header) i pozadine (engl. Background). Stavka *editor* omogućuje, uz poznavanje pravila pisanja HTML-a, CSS-a i PHP-a da se samostalno uredi stranica. Teme su vizualni dio stranice. Preuzeta tema zatim se instalira i aktivira. Izbornici se kreiraju automatski ovisno o korištenoj temi. Stavke unutar izbornika mogu se povezati sa stranicama i člancima. *Widgeti* su pomoćni programi koji se postavljaju u pomoćni stupac (engl. sidebar) unutar WordPressa. Neki su sastavni dio sustava poput arhiva, kalendara, kategorija, najnovijih komentara, postova, poveznica…), a još ih mnogo postoji na Internetu. Dodaci (engl. Plugins) omogućavaju povećanje funkcionalnosti samog WordPressa. Kroz ovaj izbornik može se upravljati postojećim dodacima, dodati nove ili promijeniti. Nakon instalacije potrebno ih je aktivirati, a većinu i dodatno podesiti. Postoji mogućnost deaktiviranja i brisanja. Korisnici (engl. Users) donose popis registriranih korisnika te omogućuje dodavanje novih ili izmjenu postojećih. Kroz ovaj sustav, korisnicima se mogu postaviti i uloge. Važno je navesti alate (engl. Tools) koji se koriste za uvoz ili izvoz *postova* i komentara iz drugih sustava. Općeniti dio (engl. General) podešava *Site Title*, *Tag Line* te datum i vrijeme stranice. Pisanje (engl. Writing) donosi postavke vezane uz uređivanje *posta* i/ili stranice poput veličine polja za pisanje. Čitanje (engl. Reading) podešava prikaz sadržaja kako ga vidi korisnik, rasprava (engl. Discussion) omogućava podešavanje dodatnih postavki vezanih uz komentare, a mediji (engl. Media) omogućavaju da se podese pretpostavljene veličine slika. Privatnost (engl. Privacy) omogućava dopuštenje tražilicama da pristupaju web stranici. Ukoliko korisnik želi da stranica bude dostupna, mora uključiti ovu opciju. Izbornik *stalne veze* (engl. Permalink) može podesiti izgled URL adrese za stalne veze i arhive (stranice, članke…). [9, 12]

WordPress je podešen tako da automatski koristi web URL-e (adrese web stranica) koji sadrže upitnike, brojke i razne druge znakove npr. *http://www.moja_domena.hr/?p=23*. Unutar WordPressa, korisnicima mogu biti dodijeljena razna prava. Primjerice, administrator

koji ima glavnu ulogu, može raditi što god želi sa sustavom.

No upravo zbog toga administrator treba biti samo jedna osoba koja za samu stranicu i odgovara. Urednik (engl. Editor) je osoba koja može pisati i objavljivati svoje članke, no može otvarati i mijenjati postojeće članke u svrhu uređivanja. Autor je osoba koja može napisati članak i objaviti ga. Suradnik (engl. Contributor) je osoba koja može napisati članak ali ga ne može objaviti. Taj njegov članak admin potvrđuje ili odbacuje. Pretplatnik (engl. Subscriber) ako stranica ima dozvolu za registracijom korisnika, nema nikakva prava mijenjanja sadržaja na samoj stranici. [12]

8. PLAN REALIZACIJE PROJEKTA

Ovaj projekt je napravljen kako bi si olakšali dnevni zadatci dizajnera, ubrzan proces izrade statičnih web stranica i implementacija istih u besplatni CMS sustav radi jednostavne dopune sadržaja. U ovom radu prikazano je kako alati, štede na vremenu, a isto tako i olakšavaju posao *frontend* dizajnera te kako je moguće bez puno znanja *backend* tehnologija napraviti dinamičnu stranicu. Radi ubrzanja procesa izrade web stranica korišten je Bootstrap *framework* za izradu statične web stranice. Pošto je Bootstrap i *mobile friendly* s njim je moguće napraviti i responzivni web dizajn (RWD). Također, postavljen je *virtualni server* na osobnom računalu i na njemu pokrenut WordPress CMS sustav na kojega je spojena vlastita Bootstrap stranica, i to dinamički napravljena pomoću WordPress funkcija. Koristeći se generatorom WordPress tema, stvorena je početna verzija WordPress teme u koju je stavljen statični sadržaj i potom pretvoren u dinamičan. Datoteka navigacije učinjena je jednostavnom radi što lakšeg korištenja, a nalazi se u *headeru* stranice. Ima nekoliko poveznica u sebi, kao npr. home, Bootstrap, WordPress, blog i kontakt. Navigacija je fiksna i pomiče se zajedno sa cijelom stranicom. Kako je i sama navigacija dinamična, ima mogućnosti dodavanja više stranica od navedenih.

Potrebna znanja i vještine su poznavanje HTML i CSS jezika te Bootstrap predefiniranih klasa, osnovno poznavanje JavaScript-a i jQuery-a, te znanje iz programskog jezika PHP i rad s bazama podataka. HTML je potreban za stvaranje samog „kostura“ stranice, zajedno sa Bootstrap klasama koje uz „kostur“ stranice odmah izgrađuje i dizajn stranice. S CSS-om, nekim elementima dodavane su drugačije vrijednosti od onih koje su definirane sa Bootstrapom te su sa pseudoklasama dodavani elementi. PHP je potreban pri izradi dinamičkog sadržaja na web stranici i za njega su korištene WordPress PHP funkcije kao *get_post_meta*, *get_field* te *bloginfo*. Sa Mysql je postavljena početna baza podataka u koju su pohranjivani, dohvaćeni i brisani sadržaji (CRUD). Pošto su korišteni WordPress *pluginovi* za olakšavanje posla, sustav se može nadograđivati sa svakom novom verzijom *plugina*, tako i sa svakom novom verzijom WordPress sustava.

9. PRAKTIČNA REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA I WORDPRESS CMS SUSTAVA

9.1. REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU BOOTSTRAP RADNOG OKVIRA

U praktičnom dijelu rada korišten je Bootstrap radni okvir s kojim je postavljena statična stranica unutar izrađenog web mjesta. Za početak, napravljena je mapa u koju se stavljaju Bootstrap datoteke potrebne za daljnji rad. Bootstrap datoteke preuzimaju se sa stranice *https://getbootstrap.com*. Nakon što se otvori stranica za preuzimanje Bootstrapa, ide se na preuzimanje (engl. Downlodad) kako bi se izabrala opcija *Compiled CSS and JS*. Zatim se, unutar stranice, naprave mape za *css* (stilski jezik), *fonts* (fontovi CSS-a), *img* (slike) i *js* (JavaScript) dokumente te se u njih pohranjuju sve preuzete Bootstrap datoteke. Unutar mape *css* pohranjuju se Bootstrap CSS datoteke te je potrebno napraviti vlastitu datoteku *mojcss.css*. U mapu *fonts* ubačene su Bootstrapove datoteke za ikone (engl. glyphicon), a u *js* mapu Bootstrapove JavaScript datoteke, dok *img* mapa ostaje za slike potrebne za stranicu. Kada je pripremljena mapa za izradu web stranice, unutar nje stvara se datoteku *index.html* koja ujedno predstavlja početnu stranicu rađenog web mjesta. U <*head*> elementu (zaglavlje HTML dokumenta), stavljene su poveznice na datoteke potrebne za funkcioniranje stranice. Bootstrap CSS datoteka povezuje se na datoteku *bootstrap.min.css*, odnosno *custom* CSS datoteku pod imenom *mojcss.css*. Nadalje, jQuery biblioteka spaja se preko izvora *googleapis* kao što je prikazano na izvornom *kodu* broj 16. Unutar <*head*> elementa potrebno je odrediti tip dokumenta i <*charset*> (skup znakova) koji se koriste na stranici zato što ih preglednik može brže čitati i prikazivati. U <*meta*> elementima određuje se veličina ekrana na kojemu se prikazuje stranica. Budući da je Bootstrap radni okvir namijenjen za sve vrste preglednika i ima predefinirane *media upite* (engl. media query), za sve veličine ekrana, u ovom je praktičnom je djelu inicijalna veličinu stavljena na 1, tako da preglednik prilikom prikaza sadržaja računa svoju širinu kao širinu ekrana na kojem se stranica pregledava. Primjerice, ako je ekran širine 1024px, a prozor od preglednika je veličine 720px, Bootstrap će prikazivati stranicu širine 720px i prema toj širini će se uključiti *media upit* za veličinu ekrana. Na kraju, potrebno je još odrediti naslov te ga staviti u <*title*> element (naslov) i time je završena izrada <*head*> elementa.

```
<head>
	<meta http-squiv=“Content-Type“ content=“text/HTML;
		charset=UTF-8“>
	<meta charset="UTF-8">
	<meta name=“viewport“ content=“width=device-width,
		initialscale=1“>
	<linkrel="stylesheet"
		type="text/CSS" href="CSS/Bootstrap.min.CSS">
	<link rel="stylesheet" type="text/CSS" href="CSS/mojCSS.CSS">
	<script
src="https://ajax.googleapis.com/ajax/libs/jQuery/3.2.1/jQuery.min.j
s"></script>
	<title>Od Bootstrapa do Worpressa</title>
</head>
```

Izvorni kod broj 16. Spajanje Bootstrapovih datoteka sa HTML dokumentom

9.1.1. HEADER

Nakon što je napravljen <*head*> element, potrebno je izraditi <*body*> dokumenta. Prvo što je potrebno napraviti je <*header*> dokumenta (zaglavlje). Prilikom spajanja Bootstrapa na WordPress, stranica se dijeli na *header*, *footer* (podnožje) i <*body*> dokumenta, posebno se piše <*header*>, a isto tako i <*footer*> dokumenta, radi lakšeg spajanja na WordPress CMS. U zaglavlju je smješten *logo*, u lijevom kutu, dok je navigacija smještena u desnom kutu. Nije potrebno pisati CSS *kod* za svaki od element navigacije. Nadalje, unutar <*header*> elementa korišten je <*nav*> element (*navbar*) s klasama *navbar, navbar-inverse, navbar-fixed-top*. S tim klasama definira se element navigacije (*navbar*), crnom bojom s bijelim slovima *navbar-inverse* te je njegova pozicija na vrhu ekrana radi praćenja pomicanja stranice *navbar-fixed-top*. Ako bi ovaj element bio iste širine kao i ekran, korisnicima ne bi bilo dobro vidljivo na velikim ekranima. Zato je unutar tog elementa napravljen još jedan <*div*> element s klasom *kontejner* (kojim se određuje veličina prikaza stranice na jako velikim ekranima). Deklaracije za klasu *kontejner* stvaraju se unutar vlastite CSS datoteke (Izvorni *kod* broj 17).

```
.kontejner{
        max-width: 1024px;
        margin:auto;
}
```

Izvorni kod broj 17. CSS deklaracija za klasu kontejner

S tom deklaracijom dobivena je najveća veličina prikaza od 1024px (max-width) i sa *margin:auto;* elementi se pozicioniraju na sredini ekrana, tako da preglednik sam određuje jednake margine s lijeve i desne strane. Nakon toga, unutar *kontejner* elementa potrebno je otvoriti element s klasom *navbar-header* u kojem se prave elementi za sve funkcije navigacije. Gumb za prikaz na manjim ekranima radi se s klasom *navbar-toggle=“collapse“* s kojom se pomoću JavaScript funkcije skriva taj isti element. S *data-target=“. navbar-collapse“* određuje se element na koji će se naredba odnositi. Sljedeća tri <*span*> elementa rade tri horizontalne linije praveći *hamburger menu* kojim daju izgled gumba za navigaciju.

```
<div class="navbar-header">
        <button type="button" class="navbar-toggle"
                data-toggle="collapse"
                data-target=". navbar-collapse">
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
               <span class="icon-bar"></span>
        </button>
. . .
```

Izvorni kod broj 18. Hamburger menu

Icon-bar klasa definirana u Bootstrap CSS-u, predstavlja blok veličine 2px i širine 22px.

```
.navbar-toggle .icon-bar {
  display: block;
  width: 22px;
  height: 2px;
  border-radius: 1px;
}
```

Izvorni kod broj 19. CSS deklaracija za icon-bar klasu

Elementi navigacije te poveznice na druge stranice, stvaraju se otvaranjem novog elementa unutar *kontejner* elementa te mu se dodjeljuju klase *collapse* (urušiti) i *navbar-collapse* s kojima se sakrivaju elementi na malim ekranima i s kojima se poziva navigacija s prethodno napravljenim gumbom *data-target=".navbar-collapse"*. Unutar toga elementa napravljena je lista koja sadržava sve poveznice navigacije. Napravljenoj listi *<ul>* dodaju se klase *nav*, da bi preglednik prepoznao da se radi o navigaciji, *navbar-nav* za pozicioniranje elementa jednog pored drugoga, a ne jedan ispod drugog, te klasa *navbar-right* koja je potrebna da se navigacija pomakne na desnu stranu ekrana, a logo na lijevu. Unutar *<ul>* napravljen je popis stavki liste sa *<li>* za svaki element navigacije i s time je završeno slaganje *<header>* elementa stranice.

```
<div class="collapse navbar-collapse">
    <ul class="nav navbar-nav navbar-right">
       <li class="active" ><a href="index.html">Home</a></li>
       <li><a href="bootstrap.html">Bootstrap</a></li>
       <li><a href="wordpress.html">WordPress</a></li>
       <li><a href="blog.html">Blog</a></li>
       <li><a href="kontakt.html">Kontakt</a></li>
    </ul>
</div>
```

Izvorni kod broj 20. Bootstrap navigacija

9.1.2. HERO IMG ZONA

Nakon navigacije, napravljen je *<hero-img>* koji predstavlja sliku, ili veliki *banner*, reklamu na početnoj stranici koja prezentira cjelokupnu stranicu. Za *<hero-img>* stvoren je element s klasom *<hero-img>* kojem su dodijeljeni CSS atributi.

```
<div class="hero-img">
       <img src="img/business-1839876_1920.jpg" class="img-
responsive"   style="background-size: cover; background-position:
center; background-repeat: no-repeat;">
</div>
```

Izvorni kod broj 21. HTML kod za hero-img dio

Važna napomena je da prilikom spajanja na WordPress, slika bude zamijenjena sa *has_post_thumbnail* funkcijom, kako bi korisnik imao mogućnost odabira vlastite slike.S Bootstrap klasom *img-responsive* slika dobiva responzivne vrijednosti i ona skalira prema veličini ekrana, tj. za svaku veličinu ekrana pozivati će određene *media upite* te se po njima stilizirati i ponašati na ekranu. Zato je u statičnoj verziji stranice bolje sliku povezati unutar HTML dokumenta i definirati joj stil, nego preko CSS klasa. U ovom slučaju unutar WordPress PHP datoteke bit će potrebno uvesti samo funkciju, a neće biti potrebno raditi posebne atribute za poziv slike.

9.1.3. MODAL

```
<div id="alert">
       <div class="row">
       <div class="kontejner">
              <div class="col-sm-8">
                     <p class="lead">
                            Lorem ipsum dolor sit amet.
                     </p>
              </div>
              <div class="col-sm-4">
              <button class="btn btn-success btn-lg btn-block"
                     data-toggle="modal" data-target="#popup">
                     Stisni da se učaniš
              </button>
              </div>
       </div>
       </div>
</div>
```

Izvorni kod broj 22. HTML kod za poziv Modala

Ispod <*hero-img*> korištena je Bootstrap JavaScript funkcija za <*modal*> (Izvorni *kod* broj 22). U *kodu* element *id=alert* definiran je u *mojcss.css* datoteci uz promjenu boje, fonta i postavljanje elementa što je prikazano u izvornom *kodu* broj 23.

```
#alert{
	background: #e74c3c;
	padding: 20px 0;
	color: white;
	p{
		margin: 10px 0 0 10px;
	}
}
```

Izvorni kod broj 23. CSS deklaracija za id="alert"

Unutar tog elementa nalazi se element s Bootstrap klasom *row*, a unutar njega, dva elementa, od kojih je jedan sa širinom stupca osam koji predstavlja *info tekst* i drugi, sa širinom stupca četiri, u kojem se nalazi gumb za poziv modala, a kojemu je dodijeljen *data-target="#popup"*. S njime se poziva modal preko ID-a. Kada je postavljen izgled *div* elementa za poziv modala, na dnu HTML dokumenta potrebno je napraviti modal (Izvorni *kod* broj 24).

```
<div class="modal fade" id="popup">
  <div class="modal-dialog">
    <div class="modal-content">
       <div class="modal-header">
         <button type="button" class="close" data-dismiss="modal">
           <span aria-hidden="true">&times;</span>
         </button>
         <h4>Učlani se!</h4>
       </div>
       <div class="modal-body">
         <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quo molestias tempora facilis sequi ipsa iure. Facere sint aliquam, hic autem. </p>
         <form class="form-inline" role="form">
          <div class="form-group">
           <label for="ime">Vaše ime</label>
           <input type="text" class="form-control" id="ime"
```

```
                placeholder="Unesite Vaše ime">
        </div>
        <div class="form-group">
          <label for="email">Vaš mail</label>
          <input type="email" class="form-control" id="email"
                placeholder="Unesite email adresu">
        </div>
        <input type="submit" class="btn btn-danger" value="submit">
       </form>
      </div>
    </div>
  </div>
</div>
```

Izvorni kod broj 24. HTML kod za izgled Modala

Element koji sadrži cijeli modal se sastoji od klase *modal* (ona uključuje JavaScript funkcije i funkciju *fade* koja predstavlja finu tranziciju između pojavljivanja i gašenja modala). Ona je povezana na Bootstrapov CSS u kojem je definirana tranzicija elementa s *transition: opacity . 15s linear;*, što u prijevodu znači da se vidljivost elementa mijenja u trajanju od 15 sekundi i da je promjena linearna (jednako mijenjanje tijekom 15 sekundi). S elementom koji ima klasu *modal-dialog* definiraju se margine i veličina modala, a *id=“popup“* povezuje modal s gumbom za pozivanje istog (Izvorni *kod* broj 25).

```
.modal-dialog {
  position: relative;
  width: auto;
  margin: 10px;
}
```

Izvorni kod broj 25. CSS deklaracije za modal-dialog klasu

Unutar *modal-dialog* (glavnog dijaloga) unesen je izgled modala koji se veže s Bootstrap klasom *modal-content* (sadržaj modala). *Modal-content* se slaže kao HTML dokument, on ima svoj *header*, *body* i *footer*. Unutar *modal-content* otvara se element s klasom *modal-header*. Sa <*button*> elementom je definiran gumb, a unutar njega <*span*>

element, odgovoran za izgled gumba. Primjerice, *&times!* je zamjena za križić u HTML *kodu*. Unutar elementa s klasom *modal-body* stavljen je generički tekst unutar <*p*> element i kontakt forma s gumbom za prijavu. Nakon napravljenog *modal-content*, potrebno je zatvoriti sve otvorene elemente kako bi napravljeni modal bio spreman za rad.

Nakon izrade <*header*>, <*hero-img*> i <*modal*>, ostatak stranice, sve do kontakt forme i *footera*, dijeli se na tri cjeline. Prva cjelina predstavlja *info prostor* podijeljen u dvije sekcije. Prva sekcija sadrži informacije o Bootstrap-u, a druga o WordPress-u. Svaka od njih podijeljena na dvije strane. Prva strana sadržava tekst dok druga sadrži responzivnu sliku. Kako bi rečeno bilo izvršeno, za cijeli odlomak, napravljen je <*div*> koji će sadržavati cijeli element. Tom *div-u* dodjeljuje se Bootstrap klasa *row* koja u *bootstrap.css* datoteci ima definirana pravila (Izvorni *kod* broj 26).

```
.row {
  margin-right: -15px;
  margin-left: -15px;
  display: table;
  content: " ";
  clear: both;
}
```

Izvorni kod broj 26. CSS deklaracija za row klasu

Sa navedenim pravilima, <*div*> s klasom *row* briše sva *float* svojstva od prijašnjih elemenata. Unutar njega stavljen je <*div*> s klasom *col-sm-8*. Ta klasa skalira na 2/3 ekrana(što znači da mu dopušta mu širenje u osam stupaca Bootstrapove mreže). U taj element unesen je *info tekst* i napravljen gumb za *„saznaj više“* poveznicu. Odmah ispod navedenog <*div*> elementa, a unutar <*div*> elementa s klasom *row*, otvara se još jedan <*div*> s klasom *col-sm-4* koji popunjava ostatak stranice, odnosno, preostalu 1/3, koja je i ostala prazna. U taj element se postavlja slika koja će biti responzivna, a napravljena takvom bit će s Bootstrap klasama kao u izvornom *kodu* broj 27.

```
<div class="col-sm-4">
       <img src="img/WordPress.jpg" class="img-responsive">
</div>
```

Izvorni kod broj 27. HTML kod za prikaz responzivne slike

Kao što se vidi u *kodu*, za element koji sadrži cijelu sliku, data je samo jedna klasa, a to je *col-sm-4* koja će element staviti preko četiri stupaca Bootstrapove mreže i ona će se odražavati na sve ekrane koji su širi od 575px, a za sve manje, elementi će se slagati jedan iznad drugoga. Unutar elementa s klasom *col-sm-4*, otvara se element za poziv slike <*img*>, a u njegov atribut *src* (engl. Source) stavljat će se URL slike.

Slike da bi se na istome mjestu učitale, u klasu elementa dodana je Bootstrap klasa za responzivne slike *img-responsive* koja sliku smanjuje i povećava ovisno o dimenziji ekrana (Izvorni *kod* broj 28).

```
.img-responsive{
  display: block;
  max-width: 100%;
  height: auto;
}
```

Izvorni kod broj 28. CSS deklaracija za img-responsive klasu

Sa završenim Bootstrap info elementom, kreće se u izradu *WordPress info* elementa koji je u suštini isti kao i *Bootstrap info* element. Iz tog razloga, kopira se cijeli *kod* elementa i ponovo lijepi uz promjenu nekoliko redova *koda*.

```
<div class="background">
 <div class="row kontejner">
       <div id="wordpress">
             <div class="wp col-sm-8">
                   <h1>WordPress</h1>
                   <p>Besplatni web software za izradu web
stranica i blogova. Program nadograđuju i održavaju stotine
softwareskih developera koji volontiraju. Najpopularniji CMS sustav
današnjice. . . <br>
                   <button type="button" class="btn btn-
danger">Saznaj više</button>
                   </p>
             </div>
             <div class="col-sm-4">
```

```
                <img src="img/WordPress.jpg" class="img-responsive">
                </div>
        </div>
 </div>
</div>
```

Izvorni kod broj 29. HTML kod za WordPress info element

U prvom redu *koda* dodan je element s klasom *background*, te se u njoj, u osobnoj CSS datoteci dodaje atribut za boju pozadine, a koja će se protezati po cijeloj širini ekrana. Nadalje, kao i u *Bootstrap info* sekciji, tako i u *WordPress*-u, počinje se s elementom klase *row*, te drugom klasom *kontejner*. Unutar tog elementa su elementi koji nose tekst i sliku. Širine su iste kao i u *Bootstrap info* (osam stupaca za tekst i četiri za sliku). Elementu za tekst dodana još jedna klasa, definirana u *custom* CSS datoteci, a to je *wp*, te je tom klasom elementu dodano *float:right;* svojstvo, radi zamjene mjesta slike i teksta, dakle, da u elementu *WordPress* prvo dolazi slika pa iza nje tekst. Sljedeća sekcija na stranici sadržava četiri jednako široka elementa, a svaki od elementa zauzima po tri stupca Bootstrapove mreže. Unutar elementa stavljene su ikone, naslov i tekst.

```
<a href="#" class="col-sm-6 col-md-3 member">
  <span class="glyphicon glyphicon-star-empty"></span>
  <h4>Mreža</h4>
  <p>Bootstrap's grid system allows up to 12 columns across the
page. </p>
</a>
```

Izvorni kod broj 30. HTML kod za element s klasom member

Elementu s klasom *col-sm-6* određeno je da će na uređajima, koji imaju zaslon ekrana manji od 992px, zauzimati širinu od šest stupaca, a s klasom *col-md-3*, da će na zaslonima većim od 992px, zauzimati po tri stupaca mreže. S klasom *memeber* u vlastitoj CSS datoteci stavlja se elementu okvir od 5px, uz bijelu boju, jer se s time optički odvajaju elementi jednog od drugog (Izvorni *kod* broj 31). S tom klasom sadržaju se također određuje boja i to s *background-color* svojstvom (Izvorni *kod* broj 30).

```
.member{
        transition-duration:600ms;
        background-color: #9B59B6;
        border: 5px solid #FFFFFF;
        text-align: center;
        color: #ffffff;
        padding: 20px;
        min-height: 240px;
        margin-bottom: 80px;
}
```

Izvorni kod broj 31. CSS deklaracija za klasu member

Stavljeno svojstvo elementa *transition-duration:600ms;* definira tranzicije koje se događaju unutar tog elementa. Tranzicija se dodaje na element s klasom *glyphicon*, koja predstavlja Bootstrapovu klasu za dohvat ikona. Unutar HTML *koda*, vidi se dodan <*span*> element s klasom *glyphicon*> i *glyphicon-star-empty* unutar elementa. Prva klasa određuje i govori pregledniku da se radi o Bootstrapovom elementu koji sadrži ikone, a drugom klasa određuje izgled te ikone. Nadalje, unutar vlastite CSS datoteke dodaje se Bootstrapovoj klasi *glyphicon* svojstvo tranzicije, a ona se događa prelaskom miša preko elementa sa ikonom (Izvorni *kod* broj 32).

```
a:hover{
  .glyphicon{
     transform:rotateY(360deg);
     transition-duration:800ms;
  }
}
```

Izvorni kod broj 32. CSS deklaracija za hover na HTML elementu <a>

S ovim *kodom* dato je svojstvo ikonama, da se prilikom prelaskom miša preko elementa, ona rotira po y osi za 360 stupnjeva, za cijeli krug, u trajanju od 800 milisekundi. Sljedeća sekcija u stranici je kontakt forma i *footer*. Kontakt forma se radi kao obična forma u HTML datoteci uz dodatak Bootstrap klase koje definiraju izgled element forme. Na izvornom *kodu* broj 33. prikazan je HTML *kod* elementa koji sadržava formu.

```
<div class="row kontejner">
       <div class="col-lg-4">
              <h3>O meni</h3>
              <p>Lorem ipsum dolor</p>
       </div>
       <form class="col-lg-8">
              <div class="form-group col-sm-6">
                  <label for="ime">Vaše ime:</label>
                  <input type="text" class="form-control" id="ime">
                </div>
                <div class="form-group col-sm-6">
                  <label for="email">Email:</label>
                  <input type="email" class="form-control"
id="email">
                </div>
                <div class="form-group col-sm-12">
                  <label for="input">Poruka:</label>
                  <textarea type="text" class="form-control input-
lg" placeholder="Vaša poruka" rows="3">
                  </textarea>
                </div>
                <button type="submit" class="btn btn-
default">Pošalji</button>
       </form>
</div>
```

Izvorni kod broj 33. HTML kod za kontakt formu

Kontakt forma kreće s elementom klase *row*. Sljedećih par redova *koda* koji su unutar elementa s klasom *col-lg-4* sadrže naslov, unutar *<h3>* elementa i paragrafa *<p>*. Ostatak prostora zauzima forma. Klasom *col-lg-8* zauzima se prostor od osam stupaca, a unutar tog prostora nalazi se kontakt forma. Kontakt formi dodijeljena je klasa *form-group* s kojom se njoj daje Bootstrap klasa za globalne stilove. Svakom od element forme, dodaju se također klase za veličine koji oni zauzimaju u prostoru forme s klasom *form-control*.

9.2. REALIZACIJA RESPONZIVNOG WEB MJESTA POMOĆU WORDPRESS CMS SUSTAVA

Kako bi statična stranica postala dinamična, potrebno ju je spojiti na CMS sustav. U ovom radu spojena je na besplatni WordPress CMS sustav za upravljanje sadržajem. Prvi korak je preuzimanje s interneta zadnje verzije WordPress CMS-a. U pregledniku se navigira do w*ordpress.org* stranice sa koje se preuzima zadnja inačica sustava.

Kada se stanica učita, u gornjem desnom kvadrantu se pojavi gumb s nazivom *Download WordPress* koji vodi na stranicu s *download* gumbom za skidanje zadnje inačice sustava. Kako bi WordPress radio na osobnom računalu, potreban je program s virtualnim serverom. U ovom je radu korišten program *Wamp*. Sa stranice *wampserver.com* skinuta je zadnja verzija wamp servera koja je instalirana na osobnom računalu. Pokrenuvši ga, u traci sa *trayicons* pojavljuje se zeleni znak za Wamp (W). WordPress, prethodno spremljen na osobno računalo, potrebno je otvoriti i spremiti u mapu od wamp servera pod nazivom *www*. Time je stvorena okolina u kojoj se može spajati Bootstrap radni okvir i WordPress CMS sustav. Radi lakšeg rada, stvorene su *starter theme*, a one su napravljene na stranici pod nazivom *underscores.me*. Ova stranica daje početnu WordPress temu koja se korigira s potrebama stranice. Naziva se kosturom stranice, daje PHP *kodove*, osnovni CSS i dr. Otvorivši stranicu *undersores.me* i kućicu s nazivom *Theme Name* otvara se mogućnost pritiska na napredne opcije (engl. Advanced Options) koje daju mogućnost da se pobliže opiše stranica. Koristeći opciju *generate*, pokreće se download osnovne WordPress teme, a pod kućicama *WooCommerce* i *_sassify!* ostavlja se prazno jer se njima koristi kod web trgovine i pri korištenju SASS generatora CSS *koda*. Kada se otvori arhiv skinut s *underscores.me* vidi se sadržaj potreban za WordPress stranicu.

Sljedeća stvar koju je potrebno napraviti, je staviti „kostur“ preuzete stranice s *underscores.me* u mapu WordPress CMS sustava koji se nalazi na *wamp* serveru. Tema se stavlja u mapu koja se nalazi u *wp-content/themes*. Time su posložene sve potrebne datoteke za pokretanje sustava i stranice. Kod instalacije WordPress-a na *wamp* serveru potrebno je napraviti bazu podataka iz koje WordPress stvara dinamički sadržaj. Baza podataka napravi se kada se u browser u URL upiše *localhost* i pokrene *phpmyadmin* i tako se napravi baza podataka za stranicu. *Username* za administratora wamp servera je *root*, a password nije potreban. Kada se uđe u *phpmyadmin*, u lijevoj traci ide se na *new* koji omogućuje rad nove baze podataka.

Nakon što se da ime bazi,napravi se administrator baze kojem se daju privilegije i ovlasti, na način da se alatnoj traci pod *privileges* stisne na *add new user*. Kada je završeno stvaranje baze podataka, daje se ime bazi podataka u *wp-config-sample.php* datoteci. Linije od 20 do 39 ispunjavaju se sa osobnim podacima koji su uneseni prilikom gradnje baze podataka. Od linije 49 do 56 definiraju se ključevi za provjeru autentičnosti. WordPress nudi generator ključeva koji se nalazi na stranici *https://api.wordpress.org/secret-key/1.1/salt/*. Kada posjetimo tu stranicu, ona automatski generira ključeve, koji se kopiraju u datoteku *wp-config-sample.php*. Ta se datoteka mora preimenovati u *wp-config.php* i time je WordPress spojen na bazu podataka. Vrativši se u preglednik, potrebno je instalirati WordPress na *localhostu*. Potrebno je ispuniti tražene podatke radi instalacije WordPressa (engl. *Install WordPress*).

Po završetku instalacije WordPressa, kopira se početna tema, preuzeta s *underscores.me* stranice, u WordPress mapu. U arhivi su preuzeti podaci te se sve datoteke premještaju u mapu WordPressa *wp-content/themes*. Temu je moguće instalirati i na drugi način, tako da se u WordPress upravljačkoj ploči pod izbornikom *Appearance* instalira tema. Aktivacija teme se radi na WordPress upravljačkoj ploči pod izbornikom *Appearance*, aktivacijom s pritiskom na ikonu *Activate*. Pritiskom na *Customize* WordPress upućuje na sučelje stranice s alatnom trakom za stiliziranje elemenata stranice, prikazuje trenutni izgled napravljene stranice ali bez CSS-a koji je oblikuje. Prikazuje samo kostur stranice kojega je moguće postaviti na željeni izgled. Nakon postavljenog servera, instalacije WordPress-a i početne teme, može se započeti sa spajanjem statične stranice na WordPress kako bi postala dinamična.

9.2.1. HEADER

Za početak spajanja Bootstrap statične stranice, potrebno je pretvoriti statični *header* u dinamički. U mapi WordPressa, gdje je spremljena tema, datoteka pod nazivom *header.php*, briše se sve ispod elementa <*div*> s klasom *hfeed site* te se kopira *header* iz statične stranice. Dinamični manu na Bootstrapom radi se pomoću predefinirane php funkcije *wp_nav_menu* koja podržava polja s kojima se pozivaju Bootstrap klase dodijeljene u statičnoj stranici (Izvorni *kod* broj 34).

```
<?php
	Wp_nav_menu(
		Array(
		'display_location => 'primary',
		'container' => 'nav',
		'container_class' => 'collapse navbar-collapse',
		'menu_class' => 'nav navbar-nav navbar-right'
		)
	);
?>
```

Izvorni kod broj 34. PHP funkcija za navigaciju

Zamijenivši statični izbornik s dinamičnim, u WordPress upravljačkoj ploči napravi se meni pod *apperance/menu* i dodaju stranice. Nadalje, ispod se kvačicom označuje *display location* i php poziva naredbe iz funkcije za meni. Da bi radile Bootstrap CSS deklaracije, u WordPress-u je potrebno povezati Bootstrap datoteke u *header.php* datoteku. Sve korištene datoteke statične web stranice kopiraju se u WordPress mapu. Iz mape *css* uzimaju se sve datoteke i kopiraju u mapu teme pod nazivom *css*, *fonts* u *fonts* i *js* u *js*, dok se *img* mapa ne treba kopirati jer se ona može izravno iz upravljačkog modula ubacivati u bazu podataka. Kako bi te datoteke radile moraju se povezati unutar *koda*, a za to se koristi WordPress PHP funkcija *<?php bloginfo('stylesheet_directory');?>* stavljena u *root* mape u kojoj se nalaze CSS datoteke. Sad kada je i CSS povezan s WordPressom počinje pretvaranje statične stranice u dinamičnu i to sa svim CSS stilizacijama koje su prije napravljene. Forma i *footer* ponavljaju se kroz cijelu stranicu, na svakoj od stranica unutar web mjesta na samome dnu su forma za upit i *footer* koji se rade u *footer.php* datoteci WordPressa.

9.2.2. FOOTER

Za spajanje *footera* na stranicu koristi se posebnu datoteku kao i za *header*. Ona je već generirana u mapi od WordPressa i samo je treba izmijeniti. Otvaranjem *footer.php*, u njega se stavlja iz statične stranice, sve od kontakt forme do kraja datoteke. Prilikom izrade stranice radi se i svaka pojedina datoteka, pa je tim putem i olakšano pisanje *footera* pošto se on pojavljuje na svakoj stranici u dizajnu. U *footeru* stranice navode se i sve JavaScript i jQuery

funkcije korištene u statičnoj stranici. Prilikom izrade modala korištena je datoteka *bootstrap.min.js* biblioteku te je potrebno spojiti je sa *bloginfo* funkcijom kao u izvornom *kodu* broj 35.

```
<script type="text/JavaScript" src="<?php
bloginfo('template_directory'); ?> /js/Bootstrap.min.js"></script>
```

Izvorni kod broj 35. PHP bloginfo funkcija

9.2.3. HOME PAGE

Prva stranica ili početna stranica home page je prva stranica koja se mora pretvoriti u dinamičnu. *Home page* ili *index* stranica podijeljena je u nekoliko segmenata. Prvi segment je *hero-img* nakon kojeg dolazi *modal*, drugi je *Bootstrap* i *WordPress info* dio, zatim *Bootstrap* i *WordPress tutorial*, nakon kojega dolazi *blog* sekcija i na kraju, *kontakt forma* i *footer*. Za početak otvara se statičnu *index* stranicu, zatim se u WordPress mapi napravi datoteku pod imenom *page-home.php* u koju se stavlja početna stranica. Da bi stranica radila i da bi ju WordPress čitao potrebno je unesti sljedeći *kod*:

```
<?php
/*
       Template Name: Home page
*/
get_header();
?>
<?php get_footer();
```

Izvorni kod broj 36. PHP početni dokument

Unutar komentara (/*.....*/) unosi se ime predloška stranice koje nudi WordPress u prikazu stranica pod izbornikom *page/attributes/template*. Funkcija *get_header* dohvaća *header.php* stranicu u kojoj je napravljen dinamični header, a *get_footer* dohvaća dinamički *footer* iz *footer.php* datoteke. Statičnu *index* stranicu, sve od *headera* pa do kontakt forme stranice kopira se u novu datoteku između *get_header* i *get_footer* funkcija. Nakon unesenog *koda* i pohrane u upravljačkoj ploči WordPressa, ide se na *pages* i odabire *add new* kako bi se stvorila novu stranicu. Kada se otvori, potrebno ju je imenovati jer će se to ime pokazivati u

glavnom meniju. Pod *page attributes* mijenja se Template na *Home page* koji dohvaća dizajn spremljen u *page-home.php* datoteku. Nakon pritiska na *update*, unose se WordPress php funkcije, a stranicu je potrebno učiniti dinamičnom.

9.2.4. HERO IMG

Spajanje *hero img* se odvija preko PHP *if has_post_thumbnail* funkcije. Na primjeru ispod je prikazan *kod* funkcije. Ova funkcija traži od WordPressa povratnu informaciju, o tome da li je u bazi podataka spremljena slika za stranicu pod *featured image* koja se postavlja u *dashboardu* WordPressa. U slučaju da ne postoji slika, da je korisnik nije odredio, WordPress uzima definiranu iz koda ispod ispod *else* naredbe.

```
<?php if ( has_post_thumbnail()) { ?>
	<?php the_post_thumbnail('post-thumbnail', ['class' => 'img-
responsive responsive--full', 'title' => 'Feature image']);
	?>
<?php } else { ?>
	<div class="hero-img" data-type="background">
		<img src="<?php
bloginfo('stylesheet_directory');?>/img/business-1839876_1920.jpg"
class="img-responsive" style="background-size: cover; background-
position: center; background-repeat: no-repeat;">
	</div>
<?php } ?>
```

Izvorni kod broj 37. PHP funkcija za prikaz Hero-img elementa

Dio koji slijedi je modal, a sljedeći segment stranice je *Bootstrap* i *WordPress info* sekcija. Unutar te sekcije sav upisani tekst u statičnoj stranici se mijenja iz upravljačke ploče. Mijenja se pritiskom na *custom fields* ili osobnim poljima uz prethodno izabranu stranicu na *Pages*. Kada se otvori ploča za upravljanje stranicom u gornjem desnom kutu, postoji opcija *postavke ekrana* (engl.Screen Options) te dodavanjem kvačice na polje *custom fields* otvora se *custom fields* opcija kojom se na stranici rade naslovi i paragrafi dinamičnima (mogu se mijenjati iz upravljačke ploče WordPressa umjesto u HTML *kodu*). Upravljački modul *custom fieldsa* sadrži polja u koja se unose podaci. Pod *Enter new* upisuje se ime polja a pod

value početni tekst koji će se prikazivati na stranici ako nije unesen bilo kakav drugi - Osnovni tekst *default*. Nakon unesenih podataka koje je *custom fields* tražio, pritisne se *Add custom fields*. U nazivu *custom fieldsa* ne mogu se koristiti zarezi, upitnici, razmaci ni ostali interpunkcijski znakovi. Nakon napravljenih *custom fieldsa* u *.php* datoteci, ispod komentara je upisan template i u njega se upisuju *custom fieldsi*.

```
$Bootstrap_info = get_post_meta ( 8, 'Bootstrap_info', true );
```

Izvorni kod broj 38. PHP get_post_meta funkcija

U ovoj funkciji *$Bootstrap_info* je varijabla s kojom se mijenjaju elementi u HTML *kodu*, a *get_post_meta* je funkcija u WordPressu koja dohvaća informacije iz baze podataka. U zagradi funkcije prvi broj označava broj stranice, ili *$post_id*, na kojoj se polje nalazi, a kako su sva napravljena polja na toj istoj stranici ide se na upravljačku ploču i otvori stranica na kojoj se radi te u url-u piše koji je broj stranice (*http://localhost/b2w/wp-admin/post.php?post=8&action=edit*) pod *post=8*. Drugi parametar koji se upisuje je *$key* ili ime polja, a to je ključ za dohvaćanje informacija, po *default* je prazan, te zadnji parametar funkcije koji vraća funkciji informaciju o količini podataka. Tako je potrebno napraviti za svaki od *custom fieldsa*, funkciju koja će dohvaćati promjene u bazi podataka. Iza toga se mora u u *body*-ju HTML dokumenta unesti varijable na mjestima koja će biti povezana s poljima.

```
<div class="row kontejner animacija">
       <div id="bootstrap">
              <div class="col-xs-8 col-sm-8 col-md-8 col-lg-8">
                     <h1>Bootstrap</h1>
                     <p><?php echo $bootstrap_info; ?> <br>
                     <a  class="btn  btn-lg  btn-danger"  href="<?php
echo  $bs_info_button_url;  ?>  "><?php  echo  $bs_info_button_name;
?></a>
                                   </p>
                     </div>
                     <div  class="col-xs-4  col-sm-4  col-md-4  col-lg-
4">
<img  src="<?php  bloginfo('stylesheet_directory');?>/img/responsive.
png" class="img-responsive">
                     </div>
```

```
            </div>
</div>
```

Izvorni kod broj 39. HTML kod s PHP funkcijama

U gore navedenom *kodu* unutar elementa *<p>* više nema *Lorem ipsum* teksta već je zamijenjen s PHP *echo* funkcijom i navedena je varijabla koja poziva funkciju napisanu na početku datoteke. Isto je napravljeno s gumbom u *<a>* elementu, koji je stavljen je unutar *href* elementa, a funkcija *echo* koja bude prikazivala *url* stranice, a koja će biti povezana unutar *$bs_info_button_url* varijable i sa *$bs_info_button_name* natpisa koji bude na gumbu. Isto se napravi sa *kodom* u kojem se mijenjaju informacije u *WordPress* dijelu stranice.

9.2.5. ADVANCED CUSTOM FIELDS

Kako bi se moglo raditi s naprednim poljima mora se instalirati *Advanced Custom Fields* (skraćeno ACF) *plugin*. Instalira se u upravljačkoj ploči pod menijem *plugins* na *add new* i upiše *advanced custom fields* te instalira *plugin* Elliota Condona, koji je besplatan i najviše korišten. Nakon instalacije ACF u upravljačkoj ploči, pod izbornikom se pojavljuje *Custom fields* u kojem se rade polja i spajaju na stranice. Ulaskom u *plugin custom fields* stisne se na *add new* kako bi bila napravljena dinamika polja. Opcije koje nudi ACF, u kućici pod brojem jedan, upisuje se ime grupe polja, a ispod, pod + *Add Field* dodaju se polja. Nadalje, ispod je kućica s lokacijom u kojoj se određuje na kojoj se stranici u *dashboardu* pojavljuju polja. Zadnja je kućica s opcijama kao broj grupe, ako postoji više grupa na jednoj stranici, pozicija, te da li se nalazi ispred, iza ili sa strane sadržaja dok je zadnja opcija namijenjena skrivanju sadržaja na ekranu.

9.2.6. ACF ADD FIELD OPCIJE

Opcija + *Add Field* nudi stvaranje novog *custom* polja te varijable koje se koriste u php datoteci. Tip polja koji omogućava odabrati vrstu polja, odnosno kakav će se sadržaj unositi u to polje, primjerice tekst, slika, link, mail adresa itd. Instrukcije polja pojavljuju se kao informacije od autora, koje pobliže objašnjavaju funkciju polja. Opcija *Required?* određuje hoće li biti obavezno ispuniti polje. *Default Value* određuje generički tekst koji će se

ispisivati na stranici ako se informacije ne unesu od strane korisnika. *Placeholder text* je tekst koji se pojavljuje unutar kućice kao podsjetnik. *Prepand* i *Append* služe da bi se moglo pisati prije ili poslije unosa. *Formating* služi za izgled na *frontend-u*, a *Caracter limit* za limit broja ili slova koja se mogu unesti u polje te *Conditional logic* koji služi napravljenim uvjetima koji moraju biti zadovoljeni za prikaz sadržaja. Kada su sva potrebna polja napravljena, u php datoteci definira se mjesto pojavljivanja polja unutar stranice. Tako se u *page-home.php* datoteci prije *get-header* funkcije, upisuju funkcije za dohvat polja. Nakon povezivanja polja s varijablama, varijable s PHP *echo* funkcijom unose u „kostur“ stranice na mjestima gdje će se prikazivati sadržaj (Izvorni *kod* broj 40).

```
<h1><?php echo $tutorial_1_name; ?></h1>
        <a href="#" class="col-xs-12 col-sm-6 col-md-3 col-lg-3 member">
                        <span class="glyphicon glyphicon-th"></span>
                        <h4><?php echo $member_1; ?></h4>
                        <p><?php echo $member_1_text; ?></p>
                </a>
                <a  href="#"  class="col-xs-12  col-sm-6  col-md-3  col-lg-3
member">
                        <span class="glyphicon glyphicon-font"></span>
                        <h4><?php echo $member_2; ?></h4>
                        <p><?php echo $member_2_text; ?></p>
                </a>
                <a  href="#"  class="col-xs-12  col-sm-6  col-md-3  col-lg-3
member">
```

Izvorni kod broj 40. PHP echo funkcije unutar HTML koda

9.2.7. POSTAVLJANJE BLOGA

Kako bi postavili *blog*, potrebno je u WordPressu napraviti novu stranicu s nazivom *Blog*. U *Settings* meniju pod opcijom *post page*, WordPressu se definira stranica na kojoj se nalaze *blog* objave. Nakon toga otvara se *index. php* datoteku u kojoj se napravi *blog* stranica. Kako bi stranica izgledala kao *blog*, iz statične stranice, iz *post.html* datoteke uzimaju se prva tri reda *bloga* i kopiraju u *index.php* datoteku odmah ispod *get_header();* funkcije te se zatvore svi otvoreni elementi (Izvorni *kod* broj 41).

```
<div class="kontejner">
	<div class="row" id="primary">
		<div id="content" class="col-sm-8">
```

Izvorni kod broj 41. HTML kod bloga

Nadalje, uzimaju se početni elementi *sidebara* <*aside*> i kopiraju ispod elementa s *id*-om *content* i klasom *col-sm-8*, a nakon toga se zatvaraju. Unutar elementa se kopira PHP funkcija <*?php get_sidebar(); ?*>. te se unutar *diva* s *id*-om *content* kopira se sve što je bilo na stranici *index.php* prije nego je započeto uređivanje iste (Izvorni *kod* broj 42).

```
<?php
	if ( have_posts() ) :
	if ( is_home() && ! is_front_page() ) : ?>
		<header>
		<h1 class="page-title screen-reader-text">
		    <?php single_post_title(); ?></h1>
		</header>
<?php
endif;
	/* Start the Loop */
while ( have_posts() ) : the_post();
	/*
	* Include the Post-Format-specific template for the content.
	* If you want to override this in a child theme, then include
a file
	* called content-___.php (where ___ is the Post Format name)
and that will be used instead.
	*/
	get_template_part(                    'template-parts/content',
get_post_format() );
endwhile;
	the_posts_navigation();
else :
	get_template_part( 'template-parts/content', 'none' );
endif; ?>
```

Izvorni kod broj 42. temple-parts.php dokument

Sada *blog* stranica ima željeni izgled, međutim, potrebno je još postaviti izgled pojedine objave na *blogu*. Otvara se u mapi *template-parts* datoteka *content.php*. i u njoj se počisti *kod* te tako izbriše nepotrebno. Tako na početku u prvoj *if* funkciji pod *the_title* mijenja se vrijednost s *h1* na *h2*, pošto je u dizajnu naslov objave veličine *h2*. Nadalje, unutar *if* funkcije *<?php if ('post' === get_post_type()) : ?>* izbriše se sve i ode u statičnu verziju stranice i kopira sa *post-details* i *post-comments-badge* (Izvorni *kod* broj 43).

```
<div class="post-details">
	<span class="glyphicon glyphicon-user">Branimir Bilić</span>
	<span class="glyphicon glyphicon-time">18. 9. 2017. </span>
</div>
<div class="post-comments-badge">
	<a	href=""><span	class="glyphicon	glyphicon-
comment"></span>168</a>
</div>
```

Izvorni kod broj 43. Statični HTML kod za detalje posta

Sada je taj sadržaj napravljen dinamičnim. U statičnoj verziji stranice upisuje se ime korisnika *(Branimir Bilić)* i vrijeme. U WordPressu, s funkcijom *<?php the_author (); ?>* i *<?php the_date();?>* napravi se da je sadržaj dinamičan i da se sam popunjava iz baze podataka (Izvorni *kod* broj 44).

```
if ( 'post' === get_post_type() ) : ?>
	<div class="post-details">
		<span class="glyphicon glyphicon-user">
			<?php the_author (); ?></span>
		<span class="glyphicon glyphicon-time">
			<?php the_date();?></span>
	</div>
	<div class="post-comments-badge">
		<a href=""><span class="glyphicon glyphicon-
comment"></span><?php comments_number(0, 1, '%'); ?></a>
	</div>
	<?php edit_post_link('edit', '<span class="glyphicon
glyphicon-pencil">', '</span'); ?>
<?php
```

```
endif; ?>
```

Izvorni kod broj 44. PHP kod za post

Kao što se vidi iznad, kod broja komentara koristila se datoteka *<?php comments_number($zero, $one, $more); ?>* . Ona prima tri argumenta, koja služe za prikaz komentara. Ono što bi ova funkcija prikazivala da nema unesenih vrijednosti je *No Comments* za *$zero*, *1 Comment* za *$one* i *% Comments* za *$more*. Pošto u dizajnu pišu samo brojke, u funkciji će biti upisano *0* za *$zero*, *1* za *$one* i *%* za *$more*. Znak *%* je zamjena za broj komentara. Nadalje, nakon informacija o autoru, stavlja se gumb za brzo editiranje s kojim se omogućuje administratoru lako izmjenjivanje sadržaja (*edit_post_link* funkcijom)(Izvorni *kod* broj 45).

```
<div>
   <?php  edit_post_link('edit',  '<span  class="glyphicon  glyphicon-
pencil">', '</span'); ?>
</div>
```

Izvorni kod broj 45. Edit_post_link PHP funkcija

Ovim funkcijama dobiven je izgled *blog* stranice, a sljedeće potrebno je urediti izgled stranice posta. U mapi WordPress-a otvara se datoteka pod imenom *single.php* i unutar njega iz statične stranice se stavljaju elementi koji definiraju stil i pozicioniranje elementa (Izvorni *kod* broj 46).

```
<div id="primary" class="content-area">
      <main id="main" class="site-main">
<div class="kontejner">
      <div class="row" id="primary">
            <div id="content" class="col-sm-8">
```

Izvorni kod broj 46. Kod single.php datoteke

Unutar njega ostave se funkcije iz datoteke *single.php*. Nakon elementa s *id=content* dodaje se element <*aside*> iz statične stranice s kojim je definiran sidebar i unutar njega stavi se php funkcija *get_sidebar* (Izvorni *kod* broj 47).

```
<aside class="col-sm-4">
	<?php get_sidebar(); ?>
</aside>
```

Izvorni kod broj 47. PHP get_sidebar funkcija

I s ovime je napravljeno da se na *single post* stranici, objava pojavi na desnoj strani sa širinom stupca od osam, a s lijeve strane će biti *sidebar* sa širinom od četiri jedinice. S ovime je završeno postavljanje *blog* objava i izgleda objava na stranici i na pojedinačnoj *blog* stranici s komentarima.

10. PREDNOSTI I NEDOSTACI PRIKAZANE METODE REALIZACIJE RESPONZIVNOG WEB MJESTA U ODNOSU NA KLASIČAN PRISTUP

WordPress je vrlo praktičan alat za realizaciju dinamičnih web stranica. On nudi nekoliko ključnih prednosti poput jednostavne i brze instalacije, dvije vrste sadržaja (stranice i postove), vrlo jednostavno planiranje i izradu predložaka, jednostavan i ugodan rad te velike mogućnosti proširenja pomoću dodataka. Na webu postoji veliki broj vodiča koji vode kroz proces izrade web mjesta i to korak po korak. To je i jedan veliki razlog zašto i sami početnici vole WordPress. Ono što ga čini posebnim je velika baza gotovih tema i dodataka (engl. Themes and Plugins). Također postoje besplatne (engl. Free) ali i plaćene teme (engl. Payed Themes). Kod dodataka (engl. Plugins), slična je situacija, tako da postoji mogućnost odabira između te dvije varijante uz mogućnost samostalnog pisanja svojih tema i dodataka. Još neke prednosti WordPress-a su da je besplatan, softver je otvorenog *koda* (Open Source), moćan i jednostavan, proširiv, trajan i prilagodljiv. Smatra se najuspješnijom CMS platformom na svijetu i na tom polju ima preko 50% udjela. Bootstrap je radni okvir razvijen s namjerom kako bi se napravio alat koji će pomoći automatizirati procese tijekom izrade web stranica, uvelike ubrzati pisanje *koda* te ostaviti prostora za inovativna dizajnerska rješenja. Prva inačica Bootstrapa sastojala se od samo CSS i HTML komponenata. Kasnije, u već veliku knjižnicu Bootstrapa mjesto nalazi i JavaScript te cijeli rad počinje težiti ka responzivnom dizajnu. U inačici Bootstrap 2.0.0. sve komponente bile su prilagođene širokom spektru mobilnih uređaja, laptopa i ekrana velikih dimenzija. Bootstrap je danas vrlo popularan radni okvir. Jedna od glavnih prednosti je što je *open source,* odnosno besplatan i dostupan svima. Korisnici mogu dodavati svoje dijelove *koda* te uzimati već postojeće i mijenjati ih prema svojim potrebama. Bootstrap uključuje responzivanu, mobilnu fluidnu mrežu koja se prilagođava u čak 12 stupaca ovisno o uređaju ili veličini početnog prikaza. Također, sastoji se od predodređenih klasa, a na službenim stranicama Bootstrapa moguće je pronaći dvije forme radnog okvira. Takve inačice spremne su za korištenje odmah.

Budući da je primjena Bootstrapa u izradi web stranica jako velika, postoji mogućnost da ako korisnici imaju već skinut Bootstrap, koji su preuzeli pregledavajući neke druge stranice temeljene na Bootstrapu, izrađena stranica će se dosta brže učitavati. Podložan je

promjenama te ga je lako prilagoditi vlastitim željama. Za razliku od klasičnog pristupa izradi web mjesta, gdje *frontend* programer piše *kod* od početka do kraja koristeći se samo svojim klasama i svojim CSS stilizacijama, svojim JavaScript funkcijama i jQuery funkcijama, sam radi *grid* po kojem radi stranicu ili radi bez mreže pa mora pisati pozicioniranja elemenata jednog po jednog, Bootstrap nam to ubrzava svojim predefiniranim klasama i CSS stilizacijama. Kako je i spomenuto, prilikom izrade navigacije u Bootstrapu, potrebno je nekoliko linija *koda* u HTML-u i nekoliko Bootstrap klasa koje su definirane u *bootstrap.min.css* datoteci, a kod klasičnog pristupa radilo bi se od samog početka sve samostalno te bi proces trajao znatno duže jer bi izrada CSS deklaracija oduzela puno više vremena. Morao bi se pisati CSS za svaki element navigacije, pozicioniranje navigacije, te pozicioniranje i izgled Loga stranice kao i ponašanje navigacije na stranici i njeno skaliranje u responzivnom dizajnu. Prilikom radnje od nekoliko linija *koda* i klasa ušteđena je ogromna količina vremena, jer ovime je više desetaka ili stotina *koda* zamijenjeno samo s nekoliko Bootstrap klasa. Ovo je naveden jedan primjer kod izrade navigacije, nadalje pozicioniranje element na stranici može biti zahtjevan posao, pogotovo ako ima puno elemenata s puno različitih pozicioniranja na stranici. Kada bi to radili klasičnim putem morali bi za svaki element pisati pozicioniranje, paziti na pozicioniranja od drugih elemenata te paziti da se ne preklapaju jedan preko drugoga i sl. Rezultat je puno linija *koda* s duljom izradom stranice. Prilikom izrade elemenata u *Bootstrap info* sekciji, sa Bootstrapovim klasama za širine stupaca su definirane veličine koliko koji element popunjava mjesta u toj sekciji. Kod klasičnog se pristupa mora pisati za svaku radnju svojstveni *kod*, za svaki od tih elemenata dodatni *kod* za sve ekrane manje od desktop računala, jer se oni ne bi skalirali niti pregledno postavljali na manjim ekranima. Također je potrebno definirati pravila za pozicioniranje elemenata na ciljanim veličinama ekrana.

11. ZAKLJUČAK

Vrativši se unazad nekoliko godina možemo se prisjetiti statičnih web stranica, web sjedišta te upornost autora koji su redovito ažurirali sadržaj tako da su FTP klijentom *uploadali* datoteke s računala na poslužitelj. Danas postoji mnogo modernih sustava za ažuriranje sadržaja pa se može birati između stotinjak rješenja, a prvi izbor za veliki broj korisnika je WordPress. WordPress je s vremenom postao svestran i sveobuhvatan alat za uređivanje sadržaja, oko kojeg se formirala velika zajednica korisnika koja svakodnevno predstavlja nove dodatke kojima se proširuju mogućnosti ovog alata. WordPress nudi nekoliko ključnih prednosti: jednostavnu i brzu instalaciju, dvije vrste sadržaja (stranice i postove), vrlo jednostavno planiranje i izradu predložaka, jednostavan i ugodan rad te velike mogućnosti proširenja pomoću dodataka. Iako je responzivni dizajn jedna od mogućnosti CSS-a, zbog svoje važnosti i sveprisutnosti zahtjeva posebnu pozornost. U novije vrijeme, korisnici su počeli pristupati web stranicama s raznih medija. Zato se javlja potreba prilagodbe web stranice uređajima s različitim rezolucijama i veličinama ekrana, odnosno re-dizajna kako bi sadržaj bio pregledan na osobnim računalima te u isto vrijeme i na drugim uređajima (npr. mobilnim). Javljaju se tipovi medija koji su korišteni za drugačiji prikaz sadržaja prilikom printanja, korištenja uređaja za Brailleovo pismo i pisače, prezentacije itd... Danas se pomoću medijskih upita dizajniraju web stranice prilagođene za svaki postojeći mobilni uređaj, tablet, laptop te bilo kakav drugi uređaj kojime se pristupa na web stranicu. Budući da se broj medija sve više povećava postaje gotovo nemoguće ispuniti sve zahtjeve korisnika, odnosno održavati web stranicu te je isto tako i učiniti prilagođenom za svaki medij. Iz tog razloga web programeri se okreću alternativnim rješenjima ovakvih i sličnih problema. Počinje razvijanje CSS radnih okvira (eng. CSS Framework) kako bi se olakšala izrada web stranica, te se smanjio utrošak vremena na pisanje jedno te istog *koda* iznova za svaki projekt.

Poanta responzivnog dizajna jest da se sadržaj prilagodi uređaju, no da se pritom ne narušava vizualni identitet stranice. Nakon što se veličina ekrana smanji, sadržaj stranice se prilagođava rezoluciji. Elementi se smanjuju za određen postotak, kao i izbornici. Kod mobilnih uređaja, elementi su razmješteni i smanjeni kako bi ostali na istom početnom prikazu, te prenijeli željenu informaciju, a pritom nisu izgubili identitet niti smisao. [5, 6]

Bootstrap postaje sastavni dio web alata i prisutan je u nizu popularnih web mjesta i aplikacija. Nadalje, dizajneri koji pretvaraju izgled web mjesta imaju mogućnosti takvih

radnji bez izmjene *kodova* ili vizualnih stilova. Granice Bootstrapa i njegovih mogućnosti se pomiču. Dizajnerima omogućuje veći doseg i „snagu". Isto tako, činjenica je da je WordPress najuspješnija CMS platforma na svijetu, za korištenje nije potreban velik opseg znanja dok su za izradu teme potrebne dvije datoteke uz vrlo jednostavno korištenje. Kao dokaz popularnosti, zanimljivo je spomenuti: „Svake sekunde objavi se 17 članaka na CMS platformi te se na Google tražilici u periodu od mjesec dana riječ *WordPress* pretraži čak 37 milijuna puta"[1].

Branimir Bilić

U Varaždinu, 11.9.2018. godine.

[1] Codeinwp: „WordPress Stats"
URL: https://www.codeinwp.com/blog/wordpress-statistics/

LITERATURA

1. J. Spurlock: Bootstrap, O'Rilley Media, Inc, California, SAD, 2013.
2. J. B. Jung: Wordpress 2.7 Cookbook, Packt Publishing Ltd., Birmingham, UK, 2009.
3. C. Peterson: Learning Responsive Web Design, O'Rilley Media, Inc, California, SAD, 2014.
4. F. Cimo: Bootstrap Programing Cookbook, Web Code Geeks, Exelixis Media P. C., 2015.
5. Tutorials Point: Bootstrap responsive web development, Tutorials point, 2014.
6. S. Bhaumik: Bootstrap Essentials, Packt Publishing Ltd. , Birmingham, UK, 2015.
7. C. Lindley: Frontend Developer Handbook 2017, Frontend masters, 2017.
8. S. F. Rahman: Jump Start Bootstrap, SitePoint Pty. Ltd. Australia, 2014.
9. B. Willians, D. Damstra, H. Stern: Professional WordPress Design and Development Second Edition, John Wiley & Sons, Inc. , Indiana, SAD, 2013.
10. J. Niederst Robbins: Learning Web Design, Fourth Edition, Littlechair, Inc. , Canada, O'Rilley Media, Inc, California, SAD, 2012.
11. https://getbootstrap.com, pristup: 20.07.2017.
12. https://developer.wordpress.org, pristup: 05.09.2017.
13. https://www.w3schools.com, pristup: 20.07.2017.
14. J. Veen: The Art & Science of Web Design, Jeffrey Veen, SAD, 2000.
15. J. Tidwell: Designing Interfaces, Second Edition, O'Rilley Media, Inc, California, SAD, 2011.
16. C. Sharkie, A. Fisher: Jump Start Responsive Web Design, SitePoint Pty. Ltd. Australia, 2013.
17. https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework), pristup: 24.07.2017.
18. https://en.wikipedia.org/wiki/Content_management_system, pristup: 08.09.2017.
19. https://en.wikipedia.org/wiki/WordPress, pristup: 08.09.2017.
20. https://en.wikipedia.org/wiki/Web_design, pristup: 24.07.2017.
21. https://mashable.com/2012/12/11/responsive-web-design/#vfChzOu0isqZ, pristup: 25.07.2017.
22. https://www.techopedia.com/definition/29569/front-end-developer, pristup: 25.07.2017.
23. https://www.codeinwp.com/blog/wordpress-statistics, pristup: 08.09.2017.

POPIS IZVORNIH KODOVA

Izvorni kod broj 1. HTML kod za izračunavanje širine ekrana 4
Izvorni kod broj 2. Bootstrap klasa row 4
Izvorni kod broj 3: Medijski upit 8
Izvorni kod broj 4: Struktura HTML dokumenta 16
Izvorni kod broj 5: CSS deklaracija 17
Izvorni kod broj 6. Početni JavaScript kod 18
Izvorni kod broj 7. Bootstrap mapa 19
Izvorni kod broj 8. Bootstrap klasa za naglašavanje 20
Izvorni kod broj 9. Bootstrap forma 21
Izvorni kod broj 10. Bootstrap horizontalna forma 22
Izvorni kod broj 11. Bootstrap glyphicon klasa 23
Izvorni kod broj 12. HTML content type 25
Izvorni kod broj 13. HTML viewport 25
Izvorni kod broj 14. HTML description 25
Izvorni kod broj 15. HTML author 25
Izvorni kod broj 16. Spajanje Bootstrapovih datoteka sa HTML dokumentom 32
Izvorni kod broj 17. CSS deklaracija za klasu kontejner 33
Izvorni kod broj 18. Hamburger menu 33
Izvorni kod broj 20. Bootstrap navigacija 34
Izvorni kod broj 21. HTML kod za hero-img dio 34
Izvorni kod broj 23. CSS deklaracija za id=alert 36
Izvorni kod broj 25. CSS deklaracije za modal-dialog klasu 37
Izvorni kod broj 27. HTML kod za prikaz responzivne slike 38
Izvorni kod broj 32. CSS deklaracija za hover na HTML elementu <a> 41
Izvorni kod broj 33. HTML kod za kontakt formu 42
Izvorni kod broj 34. PHP funkcija za navigaciju 45
Izvorni kod broj 36. PHP početni dokument 46
Izvorni kod broj 37. PHP funkcija za prikaz Hero-img elementa 47
Izvorni kod broj 38. PHP get_post_meta funkcija 48
Izvorni kod broj 42. temple-parts.php dokument 51
Izvorni kod broj 44. PHP kod za post 53
Izvorni kod broj 45. Edit_post_link PHP funkcija 53
Izvorni kod broj 47. PHP get_sidebar funkcija 54

POPIS TABLICA

Tablica 1. Media upiti za veličine preglednika [1]..... 3
Tablica 2. Veličine preglednika i klase Bootstrapa [3] 4

POPIS SLIKA

Slika 1. Elementi HTML dokumenta, izvor: codepen: „Online - path selector“[23]..... 17

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, _____Branimir Bilić_____ *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/~~ica~~ završnog/~~diplomskog~~ (*obrisati nepotrebno*) rada pod naslovom "Izrada responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava" *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
Branimir Bilić
_____
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, _____Branimir Bilić_____ *(ime i prezime)* neopozivo izjavljujem da sam suglasan/~~na~~ s javnom objavom završnog/~~diplomskog~~ (*obrisati nepotrebno*) rada pod naslovom "Izrada responzivnog web mjesta pomoću Bootstrap radnog okvira i WordPress CMS sustava" *(upisati naslov)* čiji sam autor/~~ica~~.

Student/ica:
(upisati ime i prezime)
Branimir Bilić
_____
(vlastoručni potpis)