Agilni razvoj i automatizirano testiranje web aplikacije primjenom alata Selenium

Balogović, Ivan

Master's thesis / Diplomski rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:207838

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-19**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I
INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni diplomski studij

AGILNI RAZVOJ I AUTOMATIZIRANO TESTIRANJE WEB APLIKACIJE PRIMJENOM ALATA SELENIUM

Diplomski rad

Ivan Balogović

Osijek, 2019.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 02.10.2019.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Ivan Balogović
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 824 R, 28.09.2018.
OIB studenta:	66487095329
Mentor:	Prof.dr.sc. Goran Martinović
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Doc.dr.sc. Ivica Lukić
Član Povjerenstva:	Izv. prof. dr. sc. Krešimir Nenadić
Naslov diplomskog rada:	Agilni razvoj i automatizirano testiranje web aplikacije primjenom alata Selenium
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	U teorijskom dijelu potrebno je definirati zahtjeve za moderniziranje postojeće web stranice kulturne udruge, te predložiti model nove web aplikaciju uključujući odgovarajući predložak programske arhitekture. Također, treba objasniti mogućnosti alata Selenium i potrebne pripadajuće programske tehnologije, kao i mogućnosti automatiziranog testiranja web aplikacije. U praktičnom dijelu treba provesti korisničko testiranje postojećeg web rješenja s dovoljnim brojem različitih korisnika, a prema dobivenim rezultatima planirati početak razvoja nove web aplikacije (uključujući korisničko sučelje, bazu podataka, dodatne obradbene postupke), a koristeći agilni pristup aplikaciju programski
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Vrlo dobar (4)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 2 razina
Datum prijedloga ocjene mentora:	02.10.2019.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 22.10.2019.

Ime i prezime studenta:	Ivan Balogović
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D 824 R, 28.09.2018.
Ephorus podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: Agilni razvoj i automatizirano testiranje web aplikacije primjenom alata Selenium**

izrađen pod vodstvom mentora Prof.dr.sc. Goran Martinović

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

ZAHVALA

Veliku zahvalnost dugujem svom mentoru prof.dr.sc. Goranu Martinoviću koji mi je pomogao svojim savjetima pri izradi ovog diplomskog rada, i što je imao vremena i strpljenja za moje brojne upite.

Također, zahvaljujem se kolegici Maji Klajić na svim savjetima i stručnoj podršci tijekom izrade rada, te svim kolegama iz tvrtke "Five" koji su mi na bilo koji način pružili podršku.

I na kraju, velika hvala cijeloj mojoj obitelji koja je vjerovala u mene i moj uspjeh, bez obzira na to da li se radilo o sretnim ili teškim trenucima i bez kojih sve ovo što sam do sad postigao ne bi bilo moguće.

Sadržaj

1 UVOD ... 1
1.1 Zadatak diplomskog rada ... 2
2 RAZVOJ PROGRAMSKE PODRŠKE ... 3
2.1 Metodologije programske podrške ... 3
2.1.1 Metoda vodopada ... 4
2.1.2 Metoda agilnog razvoja ... 5
2.2 Korisničko testiranje ... 9
2.2.1 Definicija korisničkog testiranja ... 9
2.2.2 Cilj korisničkog testiranja ... 10
2.2.3 Dokumentiranje korisničkog testiranja ... 11
2.2.4 Audio zapis korisničkog testiranja ... 11
2.2.5 Važnost dokumentiranja korisničkog testiranja ... 12
2.3 Pregled programskih tehnologija za razvoj weba ... 12
2.3.1 Ime domene ... 12
2.3.2 Udomljavanje weba ... 13
2.3.5 WordPress ... 15
2.3.6 Nadzorna ploča ... 17
2.3.7 WordPress tema ... 18
2.3.8 Dodaci WordPressa ... 19
2.4 Testiranje programske podrške ... 20
2.4.1 Kvaliteta programske podrške ... 20
2.4.2 Podjela testiranja programske podrške ... 21
2.4.3 Testiranje korisničkog sučelja ... 22
2.4.4 Selenium ... 23
2.4.5 JavaScript ... 26
3 KORISNIČKO TESTIRANJE ... 27
3.1 Trenutno web rješenje udruge ... 27
3.2 Korisničko testiranje web rješenja udruge ... 30
4 PROGRAMSKA PODRŠKA WEB APLIKACIJE ... 39
4.1 Registracija imena domene ... 39
4.2 Udomljavanje web stranice ... 40
4.3 Povezivanje imena domene s web poslužiteljem ... 42
4.4 Model WordPress web aplikacije ... 42
4.4.2 Razvoj novog web rješenja udruge ... 44
5 TESTIRANJE PROGRAMSKOG RJEŠENJA S ANALIZOM REZULTATA ... 58

5.1 Radno okruženje 58
5.1.1 Node.js + npm 58
5.2 Plan testiranja 59
5.2.1 EasyQA 59
5.3 WebDriver API 67
5.3.1 Implementacija 67
5.3.2 Selenium WebDriver u programskom jeziku JavaScript 68
5.4 Rezultati testova sa analizom 74
5.4.1 Allure 74
6 ZAKLJUČAK 86
LITERATURA 87
SAŽETAK 89
ABSTRACT 89
ŽIVOTOPIS 91
PRILOZI 92

1 UVOD

Tijekom razvoja bilo koje vrste digitalnog proizvoda treba biti usmjeren na krajnjeg korisnika, odnosno na korisničko iskustvo. Korisničko iskustvo je bitno zato što pokušava ispuniti korisnikove potrebe, a isto tako mora stvarati pozitivno iskustvo koje stvara vjernost korisnika prema određenom proizvodu ili usluzi. Uz razvoj programskog rješenja, testiranje predstavlja sastavni dio istog. Potrebno je osigurati jasno i jednoznačno funkcioniranje svih komponenti sustava, s fokusom na funkcionalnost, dizajn te interakciju svih pojedinih komponenata što omogućuje testiranje nakon razvoja proizvoda.

Sa sve bržim razvojem tehnologije, vidljiva su i poboljšanja svih aspekata razvoja proizvoda korištenjem programskih rješenja. Danas je digitalni identitet postao važniji nego ikad, gdje više nije važno predstavlja li on pojedinu osobu, skupinu ljudi, određeni brand, ili nešto drugo. Iz tog razloga je izuzetno važno da taj identitet šalje jasnu poruku, te da bude u koraku sa sve bržim razvojem tehnologije i zahtjevima korisnika. Cilj ovog rada je izraditi novo, modernizirano web rješenje udruge "Red vitezova Ružice grada" iz Orahovice, te izvršiti skup automatiziranih testova kojima je cilj provjeriti funkcionalnost web rješenja.

Drugo poglavlje obuhvaća pregled te definiranje svih tehnologija i alata korištenih za izradu diplomskog rada, odnosno teorijski pregled korištenih tehnologija i metodologija gdje je kao uvod predstavljen koncept programskog razvoja. Nadalje, definiran je proces korisničkog testiranja pomoću kojega je potrebno dobiti uvid u prednosti i nedostatke trenutnog web rješenja. Na temelju te analize, izradit će se novo web rješenje. Također je potrebno je definirati, te predstaviti bitne značajke korištenih tehnologija za izradu web rješenja, te na kraju poglavlja prikazati alate korištene za automatizirano testiranje. Svrha trećeg poglavlja je prikazati izgled i funkcionalnosti starog web rješenja udruge, a nakon toga predstaviti podatke koji su dobiveni korisničkim testiranjem. Glavni cilj četvrtog poglavlja je prikazati način izrade web rješenje, te koje su tehnologije korištene. Dakle, prikazan je način odabira servisa koji predstavlja registar domena na kojemu je ime domene za web rješenje registrirano, servisa koji pruža usluge udomljavanja web stranica, te način na koji su iste povezane s platformom WordPress. Nakon toga, prikazan je postupak izrade novog web rješenja. Peto poglavlje prikazuje provedbu automatiziranog testiranja nad novim web rješenjem i analizu rezultata testiranja.

1.1 Zadatak diplomskog rada

U teorijskom dijelu potrebno je definirati zahtjeve za moderniziranje postojeće web stranice kulturne udruge, te predložiti model nove web aplikacije uključujući odgovarajući predložak programske arhitekture. Također, treba objasniti mogućnosti alata Selenium i potrebne pripadajuće programske tehnologije, kao i mogućnosti automatiziranog testiranja web aplikacije. U praktičnom dijelu treba provesti korisničko testiranje postojećeg web rješenja s dovoljnim brojem različitih korisnika, a prema dobivenim rezultatima planirati početak razvoja nove web aplikacije (uključujući korisničko sučelje, bazu podataka, dodatne obradbene postupke), a koristeći agilni pristup aplikaciju programski ostvariti prema zahtjevima korisnika. Također, treba napraviti plan testiranja, definirati testne scenarije, te napisati i provesti testove izrađene web aplikacije. Rezultate testova potrebno je prikladno analizirati i interpretirati.

2 RAZVOJ PROGRAMSKE PODRŠKE

Tvrtke danas djeluju u globalnom okruženju, te im je moguće pristupiti iz bilo kojeg kraja svijeta, zahvaljujući razvoju interneta kao takvog. Potrebno je posjedovati digitalni proizvod kao odgovor izazovima koje se nude na tržištu te biti svjestan da uz novo tržište dolazi do pojave konkurentskih proizvoda i usluga koje nude druge tvrtke koje imaju istog ciljanog korisnika. Prema [1], programska podrška predstavlja dio svakog poslovnog okruženja te iz toga proizlazi da je potreban način brzog razvijanja istog kako bi se iskoristile određene prilike na tržištu koje su dostupne za proširenje poslovanja. Dakle, potrebni su što brži i kvalitetniji razvoj te isporuka za što bolju konkurentnost na tržištu, gdje što bolja vidljivost (eng. visibility) predstavlja priliku da potencijalni klijenti primijete proizvod koji je njima i namijenjen.

2.1 Metodologije programske podrške

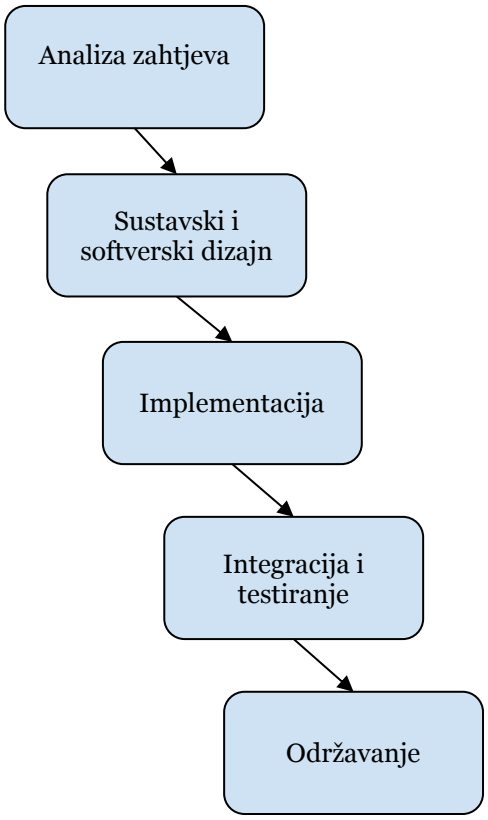
Proces razvoja programske podrške je set povezanih aktivnosti koje vode prema konačnoj proizvodnji programskog sustava. Postoji veliki broj različitih vrsta programskog sustava, gdje ne postoji neka univerzalna metoda programskog inžinjerstva koja je primjenjiva na sve. Prema tome, ne postoji univerzalno primjenjiv proces programske podrške. Iako postoji mnogo različitih procesa koji se moraju provesti tijekom samog razvoja svi oni moraju u nekom obliku sadržavati četiri temeljne aktivnosti:

- Specifikacija programa - Funkcionalnost programa i ograničenja njegova rada moraju biti definirana.
- Razvoj programske podrške - Mora se izraditi program koji zadovoljava specifikacije.
- Provjera valjanosti programske podrške - Programsko rješenje je potrebno testirati kako bi se osiguralo da radi ono što korisnik želi.
- Životni vijek programske podrške - Program se mora razvijati u skladu s promjenjivim potrebama korisnika.

Ove aktivnosti su same po sebi kompleksne te u sebi sadrže manje aktivnosti (u smislu manjih zadataka) koje služe za što bolji i što jednostavniji način upravljanja projektom. Uzevši u obzir promjenjive zahtjeve i potrebe klijenata, ključan je odabir odgovarajuće razvojne metodologije. Dvije su vodeće i najučestalije metodologije razvoja programske podrške, a to su metoda vodopada i metoda agilnog razvoja.

2.1.1 Metoda vodopada

Prema [2], metoda vodopada je osnovni model životnog ciklusa razvoja programske podrške. Popularnost ovog modela razvoja je prije bila popularnija nego što je to danas. No bez obzira na tu činjenicu, metodu vodopada je važno spomenuti zato što se svi ostali modeli temelje na njoj. Prema modelu vodopada, životni ciklus razvoja programske podrške dijeli se na nekoliko faza te podržava pretpostavku da sljedeća faza može početi nakon što je završena prijašnja faza. Odnosno, izlaz jedne faze će biti ulaz u drugu fazu. Metoda vodopada je poznata i pod nazivom linearni sekvencijalni model životnog ciklusa. Ovdje se faze vremenski ne preklapaju. Slikovni prikaz modela vodopada je prikazan na slici 2.1, a faze su preuzete iz [1].



Slika 2.1 Model metode vodopada

Faze u modelu vodopada izravno odražavaju temeljne aktivnosti razvoja programske podrške. *Analiza zahtjeva* predstavlja mogućnosti, ograničenja i ciljeve sustava koji se uspostavljaju savjetovanjem s korisnicima sustava. Potrebno je napraviti detaljno istraživanje zahtjeva klijenata te znati popis svih funkcionalnosti koje konačna verzija sustava treba sadržavati.

Sustavni i softverski dizajn se definira u sljedećoj fazi, gdje prema [1] on uspostavlja cjelokupnu arhitekturu sustava. Potrebno je proučiti koji su zahtjevi doneseni u prethodnoj fazi od strane klijenta te na temelju toga doći do dizajna cjelokupnog rješenja.

Treću fazu u modelu vodopada predstavlja *implementacija* tijekom koje se dizajn programskog rješenja realizira kao skup programa odnosno programskih jedinica. Drugim riječima, dizajn se u ovoj fazi provodi u djelo.

Integracija i testiranje predstavlja fazu gdje su pojedinačne programske cjeline ili programi integrirani i testirani kao cjeloviti sustav kako bi se osiguralo ispunjavanje programskih zahtjeva. Nakon testiranja, program se isporučuje kupcu, odnosno klijentu. Bitno je napomenuti da testiranje predstavlja konačnu provjeru prije izlaska proizvoda na tržište te samim time smanjuje vjerojatnost da taj proizvod ima veliki broj grešaka nakon izlaska na tržište.

Kako bi sustav bio funkcionalan i nakon isporuke klijentu, potrebno ga je održavati, a to je obično najduža faza životnog ciklusa. Sustav je implementiran i stavljen u praktičnu uporabu, a održavanje uključuje ispravljanje pogrešaka koje nisu otkrivene u ranim fazama životnog ciklusa. Tijekom završne faze životnog ciklusa (rad i održavanje) program je stavljen u uporabu. Nakon određenog vremena dolazi do otkrivanja pogrešaka i propusta u početnim zahtjevima programske podrške, te nakon toga dolazi do potrebe da se razvijaju nove funkcionalnosti te se sustav stoga mora naknadno razvijati kako bi ostao koristan. No izrada tih promjena (održavanje programa) može uključivati ponavljanje prethodnih faza procesa. Dakle, ako se određeni sustav razvija prema modelu vodopada, rješenje s kojim je moguće raditi ne pojavljuje se sve do kraja cjelokupnog procesa. To bi značilo da klijent dobije gotov proizvod tek nakon faze testiranja, gdje ga dobije bez mogućnosti izmjene onoga što je već implementirano. Nakon uzimanja u obzir svih činjenica o procesu izrade sustava prema metodi vodopada dolazi se do zaključka da je ta metoda neprikladna jer ne pruža fleksibilnost prema zahtjevima klijenata tijekom procesa implementacije. Isto tako nije pogodna ako klijent odluči da bi jedan dio sustava bilo dobro drugačije implementirati ili ispraviti te ga prilagođavati potrebama tržišta i ciljanoj korisničkoj skupini kroz vrijeme nakon razvoja. Do dodatnog problema može doći ako klijent traži drugačiju iskoristivost pojedine funkcionalnosti sustava, a to metoda vodopada ne može pružiti jer je kod nje naglašeno čekanje gotovog proizvoda s razrađenom potpunom funkcionalnošću.

2.1.2 Metoda agilnog razvoja

Pitanje na koje je potrebno dati odgovor je "Što je agilno?". Prema [3], agilnost je sposobnost stvaranja i reagiranja na promjene, to je način rješavanja neizvjesnog i turbulentnog okruženja i konačno postizanja uspjeha. Autori agilnog manifesta odabrali su agilnost (eng. Agile) kao oznaku za cijelu ideju jer ta riječ predstavlja adaptivnost i odgovor na promjene koje su bile tako važne za njihov pristup. U stvari se radi o razmišljanju što se događa u okruženju u kojem se nalazimo, gdje

je potrebno identificirati koji su problemi s kojima se moguće susresti, te na koji način se možemo istima prilagoditi kroz vrijeme.

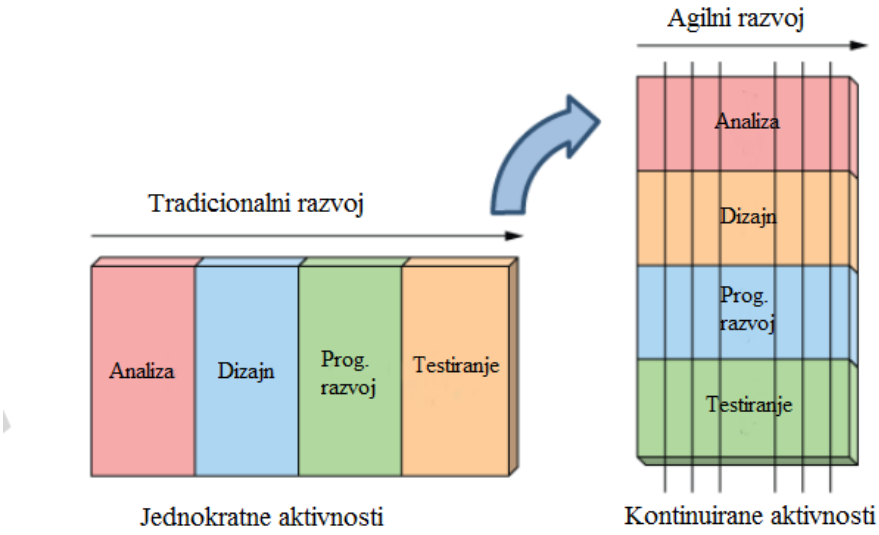
Budući da se internet zadnjih nekoliko desetljeća tako brzo razvio, lako je zaboraviti da World Wide Web još nije star ni 30 godina. Prema [4], tijekom početnih faza razvoja različitih web rješenja, dizajneri su se borili s osnovnom zadaćom prevođenja vrsta informacija koje su pronašli u knjigama na web-lokaciju. Kod ilustracije ideja koristili su papirne modele, okvire žica i dijagrame tijeka, različiti scenariji korištenja i fokus grupe bili su glavni izvor povratnih informacija koje su se mogle iskoristiti za daljnji razvoj. Razvojni inženjeri uskoro su otkrili da izrada programa ne bi trebala biti potpuno sekvencijalni proces s obzirom na to da uvijek postoje neočekivane pogreške, kao i nove tehnološke prepreke koje treba prevladati, a predviđanje zahtjeva korisnika postalo je znanost sama po sebi. Srećom, sada kada je veliki broj osoba u svijetu tako dobro međusobno povezan, gdje je lako prikupiti povratne informacije korisnika u stvarnom vremenu, tako da je moguće bolje razumijevanje načina na koji korisnici komuniciraju s web aplikacijama.

Povećanje broja korisnika na internetu razvija se u skladu s napretkom sustava za upravljanje sadržajem, što je omogućilo većem broju ljudi mogućnost da dizajnira i uredi web stranicu gdje nije potrebno toliko znanja o web tehnologijama kao prije. Budući da je proces izrade web aplikacije postao mnogo jednostavniji nego što je to prije bio slučaj, programeri su se usredotočili na usavršavanje svojih znanja kako bi odgovorili na rastuće potrebe korisnika i potrošača.

Brzi razvoj programske podrške postao je poznat kao agilni razvoj ili razvoj koji je moguće ostvariti pomoću agilnih metoda. Te metode su i osmišljene da služe za učinkovito stvaranje programskih aplikacija koje su spremne za korištenje i koje je jednostavno održavati. Sve agilne metode koje su predložene dijele niz zajedničkih značajki:

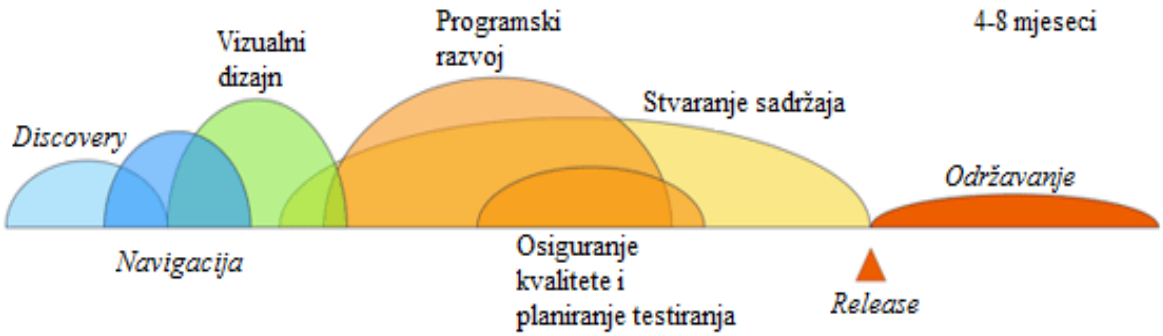
- Procesi koji se odvijaju tijekom specifikacije, dizajna i implementacije su isprepleteni. Ne postoji detaljna specifikacija sustava, a projektna dokumentacija se automatski minimizira ili generira programskim okruženjem koje se koristi za implementaciju sustava. Dokument koji se sastoji od zahtjeva korisnika je okvirna definicija najvažnijih karakteristika sustava.
- Sustav se razvija u nizu prirasta gdje su krajnji korisnici i ostali sudionici sustava uključeni u određivanje i procjenu svakog prirasta. Oni mogu predložiti promjene u programskom rješenju i nove zahtjeve koji bi trebali biti implementirani u novoj verziji sustava.
- Opsežna podrška alata koristi se kao podrška razvojnom procesu. Alati koji se mogu koristiti uključuju alate za automatizirano testiranje, alate za podršku kod upravljanja konfiguracijom i integracijom sustava te alate za automatizaciju proizvodnje korisničkog sučelja.

Prema [4], analiza, dizajn, razvoj i testiranje su aktivnosti koje se provode kontinuirano. Na slici 2.2 se nalaze razlike između tradicionalnog i agilnog razvoja, a napravljena je prema [4].



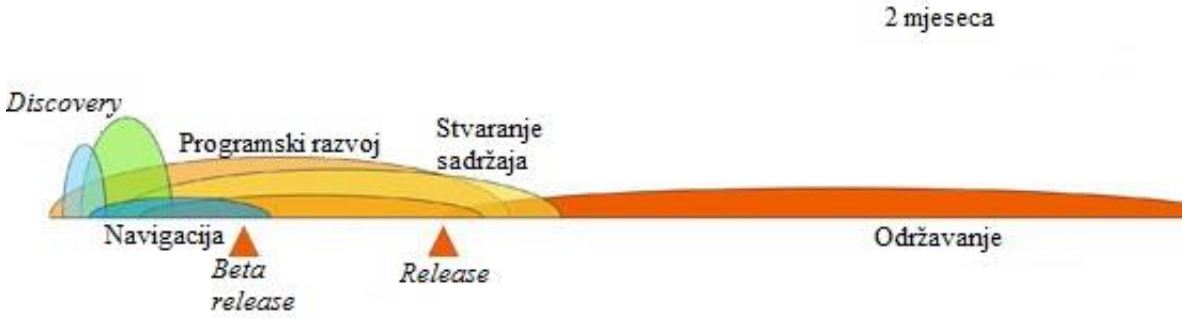
Slika 2.2 Razlika između tradicionalnog i agilnog razvoja programske podrške

Agilni razvoj označava drugačiji način razvoja digitalnog proizvoda, gdje je potrebno ukloniti nepotrebne procese koji oduzimaju vrijeme, također je potrebno što češće provjeravati funkcionalnosti proizvoda te biti siguran da se što manje vremena provodi na stvari koje ne dodaju vrijednost cjelokupnom projektu. Sve se svodi na to da je potrebno dobro rasporediti vrijeme koje je predviđeno za razvijanje određenog proizvoda gdje je vrijeme potrebno usmjeriti na radnje koje podižu proizvod na neku veću razinu u smislu kvalitete i jednostavnosti.
Slika 2.3 prikazuje tradicionalni proces izrade web aplikacije, a napravljena je prema [5].



Slika 2.3 Tradicionalni proces razvoja programske aplikacije

Kao što je vidljivo na slici, dijelovi razvojnog procesa se preklapaju u različitim vremenskim periodima razvoja, ali postoji jasan slijed koraka razvoja proizvoda. Agilni razvoj, s druge strane, smatra da većina koraka razvoja traje i istovremeno, što je prikazano na slici 2.4.



Slika 2.4 Agilni proces razvoja programske aplikacije

Najveća razlika je u tome što su inženjeri uključeni u sadržaj i navigaciju od početka projekta, pojedinačni izazovi rješavaju se kako nastaju umjesto da se čeka na velike promjene prije pokretanja. Beta verzija aplikacije (eng. Beta release) također je postala standardna kako bi se počelo prikupljati povratne informacije što je prije moguće, a održavanje se smatra jednako važnim za sve ostale korake. Održavanje je ovdje veoma bitno i moćno ako se provodi u konstantnoj komunikaciji s klijentom. Za razliku od prije spomenute metode vodopada, u agilnoj je metodi moguće promijeniti pojedine funkcionalnosti sustava bez da se prolazi kroz sve faze razvoja. Filozofija metode agilnog razvoja odražava se u *agilnom manifestu* koji su izdali vodeći programeri tih metoda. Manifesto nalaže sljedeće [6]:

- Uključivanje klijenta u proces - Klijenti bi trebali biti uključeni u proces razvoja. Njihova je uloga pružiti i odrediti prioritetne zahtjeve novog sustava te procijeniti ponavljanje sustava.
- Prihvaćanje promjena - Potrebno je očekivati da će se zahtjevi sustava promijeniti te je zbog toga potrebno dizajnirati sustav koji se može prilagoditi tim promjenama.
- Inkrementalna isporuka – Programsko rješenje se razvija u koracima, a kupac navodi zahtjeve koji će biti uključeni u svaki korak.
- Održavanje jednostavnosti - Potrebno se usredotočiti na jednostavnost u programskoj podršci i u procesu dizajna. Kad god je to moguće, potrebno je aktivno raditi da se smanji kompleksnost sustava.
- Ljudi, ne procesi - Potrebno je prepoznati i iskoristiti vještine razvojnog tima. Članove tima treba ostaviti da razvijaju vlastite načine rada bez propisanih procesa i pravila.

Nakon prikaza te analize metode vodopada i metode agilnog razvoja programske podrške zaključak je da je razvoj agilnom metodom pogodniji za razvoj novog web rješenja udruge. Prva prednost agilnog razvoja je ta da klijent ima česte i rane prilike vidjeti na koji se način razvija programska podrška, te samim time može sugerirati osobi koja razvija proizvod što bi sve trebalo prilagoditi. Ako je klijent uključen u donošenje odluka tijekom procesa razvoja, dobije osjećaj vlasništva i samim time je zadovoljniji kod konačne isporuke proizvoda. Sam klijent također mora biti svjestan činjenice da on često vidi proizvod u tijeku razvoja, koji je kod svakog sastanka nedovršen, ali to je dodana vrijednost metode agilnog razvoja u smislu transparentnosti cijelog procesa. Omogućujući klijentu da odredi koje značajke sustava imaju najveći prioritet, osoba koja razvija proizvod ima jasnu predodžbu što je klijentu najvažnije te samim time može ispostaviti samo ono najvažnije i tako osigurati što veću kvalitetu proizvoda. Da bi se isporučio što bolji proizvod, potrebno je poduzeti sve moguće mjere koje bi mogle pridonijeti stvaranju toga cilja.

2.2 Korisničko testiranje

Korisničko testiranje bi trebalo izvršiti tijekom razvoja svakog projekta, odnosno proizvoda. Razlog tome je što čak ni najbolji dizajneri na svijetu ne mogu pretpostaviti svaku radnju koju će korisnik izvršiti, dizajneri imaju ideje koje se temelje na najboljoj praksi te iskustvu, a proces testiranja omogućuje potvrdu tih ideja. Korisničko testiranje je izrazito bitno, a glavni razlog je smanjenje rizika gdje u skoro svakom pogledu poboljšava proces razvoja programske podrške.

2.2.1 Definicija korisničkog testiranja

Korisničko testiranje predstavlja proces pomoću kojeg je moguće dobiti povratnu informaciju o određenom proizvodu, značajki ili prototipu na temelju uvida stvarnih korisnika. Postoji nekoliko razloga zašto je korisno proći kroz proces korisničkog testiranja. Recimo ako je proizvod trenutno u procesu razvijanja testiranje omogućuje dizajnerskom timu da identificira jedan dio korisničkog iskustva čija funkcionalnost možda i nije logična korisniku pa ga je moguće prilagoditi tako da zadovoljava korisnikove zahtjeve [7]. Moguć je slučaj gdje određena grupa ljudi posjeduje vlastito digitalno rješenje, te dolazi do zaključka da im treba novo. Nakon postizanja odluke da se programski razvije novo rješenje, potrebno je odrediti na koji način bi taj proces bilo najbolje ostvariti. Prvi korak može predstavljati korisničko testiranje s ciljem utvrđivanja što korisnicima nije odgovaralo kod starog programskog rješenja, gdje bi u tom slučaju bilo potrebno izvršiti proces korisničkog testiranja nad starim web rješenjem s ciljem dobivanja uvida u korisničko iskustvo koje je moguće upotrijebiti kod razvijanja novog web rješenja. Podaci korisničkog

testiranja predstavljaju moćno sredstvo koje nudi jasne smjernice za razvoj samog proizvoda. Priprema korisničkog testiranja uključuje kreiranje testnog plana, okupljanje korisnika koji će sudjelovati u procesu testiranja, te na kraju realizaciju testa uz analizu rezultata da bi se dobila neka saznanja i preporuke za daljnji razvoj. Potrebno je slijediti sljedeće upute za dobivanje vrijedne povratne informacije od korisnika tijekom samog korisničkog testiranja:

- Korisnici moraju predstavljati stvarnu korisničku bazu proizvoda ili usluge.
- Test mora biti nepristran, tako da se na radnje korisnika ne utječe na načine koji dovode do rezultata pristranosti.
- Testiranje je dokumentirano ili zabilježeno za kasniji pregled.

Općenito se testiranje može učiniti u biti u bilo kojoj fazi razvoja. Nije potrebno čekati funkcionalni prototip proizvoda. Ako je potrebno, moguće je početi s testiranjem ideja na papiru ako je to relevantno. Općenito govoreći, što se prije počne s procesom testiranja tijekom projektiranja, kasnije ima manje prepravljanja. U idealnim uvjetima bi trebalo provesti nekoliko testiranja tijekom životnog vijeka projekta, ali samo ako ima vremena za to i naravno, ako budžet do dopušta. Korisničko testiranje proizvoda ili programa donosi kvalitetnu provjeru i potvrdu da su svi zahtjevi ispunjeni što pruža veću vrijednost klijentima u smislu financija, kvalitete, performansi proizvoda, te na kraju zadovoljstva korisnika koji koriste proizvod.

2.2.2 Cilj korisničkog testiranja

Iako nije potpuno precizirano koliko vremena i truda treba uložiti, te koji pristup koristiti, osoba koja provodi testiranje trebala bi prikupiti informacije od korisnika da bi podržala neku premisu koja joj služi za što jednostavnije i učinkovitije kreiranje proizvoda ili usluge. Da bi testiranje bilo učinkovito i precizno potrebno je poduzeti određene korake:

- Dobro istražiti mišljenje korisnika o proizvodu, idealno u njihovom osobnom kontekstu.
- Istražiti ne samo njihovo ponašanje, nego i značenje iza toga ponašanja.
- Korištenjem tih uvida ukazati na što bolji dizajn, uslugu, proizvod ili na neko drugo rješenje [8].

Cilj je dobivanje što boljeg uvida u korisničko iskustvo o određenom proizvodu kako bi saznali nešto novo, saznanja koja se dobiju od istraživanja ne samo da usmjeravaju proces dizajna proizvoda, već ga i inspiriraju. U dosta slučajeva cilj testiranja korisničkog iskustva koje korisnici imaju o određenom proizvodu predstavlja proces otkrivanja koje su potrebe korisnika, a testiranje se provodi zato što postoji mogućnost da inženjeri koji se bave procesom izrade proizvoda nemaju

uvid u stvarne potrebe korisnika te se uvijek može pojaviti jedan dio proizvoda ili korisničkog iskustva koji nije zadovoljio sve korisnikove potrebe i želje.
Razlozi zbog kojih korisničko testiranje može biti značajno:

- Kao način prepoznavanja novih mogućnosti razvoja proizvoda, prije nego što je poznato što može biti dizajnirano.
- Precizirati hipoteze dizajna, nakon što su poznate neke od ideja što će biti dizajnirano.
- Redizajn i ponovno pokretanje postojećih proizvoda i usluga, kada postoji povijest postojanja proizvoda na tržištu [8].

Povratna informacija koju je moguće dobiti od korisnika je veoma bitna stavka u procesu razvoja određenog proizvoda. Uvid koji se dobije tijekom procesa razgovora s korisnikom te slušanja korisnika za vrijeme njihovog korištenja određenog proizvoda pruža mogućnost izrade i dizajna proizvoda koji korisnici stvarno žele koristiti, te u isto vrijeme pruža mogućnost da se ostane korak ispred konkurencije.

2.2.3 Dokumentiranje korisničkog testiranja

Dokumentiranje je način na koji se bilježi konačan, detaljan zapis razgovora kojeg je potrebno provesti u procesu razgovora s korisnikom da bi se dobili podaci korisničkog iskustva u odnosu na određeni proizvod. Nadalje, to je način na koji osoba koja provodi razgovor s korisnikom na što učinkovitiji način bilježi podatke i koristi ih u kasnijem procesu izrade samog proizvoda. U postupku intervjua, treba ostati angažiran s korisnikom i izvući što više informacija. Izuzev toga, dokumentacija također usmjerava sam intervju u smislu stvaranja priče kroz podatke. Izrada dokumentacije tijekom razgovora s korisnikom predstavlja važan uvid za daljnji razvoj proizvoda. Zaključak je da bi valjalo snimiti (audio ili video) proces korisničkog testiranja, a to je u kombinaciji s vođenjem bilježaka najbolji način da se sačuvaju sve bitne informacije. Određen broj ljudi smatra da im zapisivanje, odnosno vođenje bilježaka pomaže filtrirati i naposljetku bolje zapamtiti sve informacije koje se spomenu tijekom samog razgovora s korisnikom, koje je kasnije moguće iskoristiti [8].

2.2.4 Audio zapis korisničkog testiranja

Audio snimanje obuhvaća sve verbalne interakcije između osobe koji provodi testiranje i korisnika koji je subjekt samog korisničkog testiranja. Najjednostavniji način bilježenja procesa korisničkog testiranja je koristiti ugrađenu aplikaciju za snimanje zvuka na pametnom telefonu, ali prvo je valja testirati da se procjeni kvaliteta same snimke. Nakon završetka je potrebno napraviti transkript

razgovora, odnosno napraviti potrebnu dokumentaciju, te zabilježiti o kojem se korisniku radi što uz vođenje osobnih bilježaka na papiru predstavlja najbolji način dokumentacije korisničkog testiranja.

2.2.5 Važnost dokumentiranja korisničkog testiranja

Dokumentacijom se zabilježava detaljan zapis intervjua, što predstavlja način da se obradi i zapamti proces korisničkog testiranja nad određenim proizvodom.
Prema [8], razlozi za dokumentaciju su sljedeći:

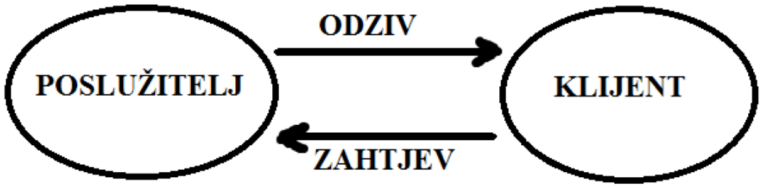
- Nije moguće dobiti sve podatke samo zapisivanjem bilješki. Da bi zabilježili sve što je korisnik primijetio kod proizvoda potrebno je i zapisivati njegove dojmove (verbalne i neverbalne) te snimati audio zapis cijelog intervjua, inače se mogu dogoditi propusti u pogledu zabilježavanja svih stavki razgovora.
- Snimanje zvuka ili video zapisa je jedini način za snimanje svih detalja intervjua.
- Potrebno je održavati intervju aktivnim te postavljati pitanja korisniku.

2.3 Pregled programskih tehnologija za razvoj weba

2.3.1 Ime domene

Ime domene je adresa određene web stranice, odnosno URL (eng. Uniform Resource Locator) poveznica koja predstavlja jedinstveno ime određenog web rješenja. Imena domena djeluju kao nadopuna ili zamjena za IP (eng. Internet Protocol) adresu web lokacije, gdje IP adresa predstavlja jedinstveni niz brojeva koji identificira računalni poslužitelj s ciljem pronalaska lokacije na kojoj se nalazi web stranica. Unutar URL-a domene, nalazi se resurs koji uključuje svaki podatak koji računalo treba da bi pronašlo određenu stranicu, dio stranice ili dokument na istoj [9]. Zbog što lakšeg korištenja, a da ne bi bile potrebe za pamćenjem cijele IP adrese, donesena je odluka da se svaka pojedina web stranica označi jedinstvenim imenom domene. Zahvaljujući imenima domena moguće je posjetiti određenu web stranicu unošenjem jedinstvenog imena domene u web tražilicu, gdje on šalje zahtjev sustavu domena odnosno DNS-u (eng. Domain Name System), koji je poput adresara za web stranice. Kada se web adresa upiše u tražilicu, on šalje zahtjev koji provjerava DNS kako bi pronašao stvarnu adresu web stranice prije nego što može pristupiti podacima na istoj. Web tražilica mora otkriti na kojem je poslužitelju web stranica udomljena te nakon toga može slati HTTP (eng. Hypertext Transfer Protocol) zahtjev na pravo mjesto. HTTP je protokol koji definira pravila za komunikaciju između klijenta i poslužitelja. Prema [10], računala spojena

na mrežu nazivaju se klijenti i poslužitelji. Pojednostavljeni dijagram načina na koji oni komuniciraju vidljiv je na slici 2.5, koja je napravljena prema [10].



Slika 2.5 Komunikacija klijenta i poslužitelja

Klijent je uređaj koji je povezan na Internet, gdje taj uređaj može biti osobno računalo te program za pristup internetu dostupan na tim uređajima, odnosno web tražilica. Poslužitelji su računala koja udomljavaju web stranice, web mjesta ili web aplikacije. Kad klijentski uređaj želi pristupiti web stranici, kopija web stranice preuzima se s poslužitelja na klijentski uređaj kako bi se prikazala u korisničkoj web tražilici [10].

2.3.2 Udomljavanje weba

Udomljavanje weba (eng. web hosting) je usluga koja omogućuje objavljivanje web stranice ili web aplikacije na internetu. Nakon prijave za uslugu web hostinga, iznajmljuje se dio prostora na poslužitelju na koji je moguće pohraniti sve datoteke i podatke potrebne za pravilno funkcioniranje određene web stranice. Poslužitelj je fizičko računalo koje radi bez ikakvih prekida tako da je web stranica dostupna svim korisnicima koji je žele posjetiti. Tvrtka koja omogućuje uslugu udomljavanja (eng. hosting) odgovorna je za održavanje i pokretanje poslužitelja, zaštitu od zlonamjernih napada i prijenos vlastitog sadržaja (tekst, slike, datoteke) s poslužitelja u preglednike posjetitelja. Kada je potrebno razviti novu web stranicu, potrebno je pronaći tvrtku koja pruža prostor na poslužitelju, odnosno pruža usluge udomljavanja. Tvrtka koja pruža uslugu udomljavanja pohranjuje sve datoteke na poslužitelju, a kada korisnik upiše ime domene u adresnu traku web tražilice poslužitelj prenosi sve datoteke potrebne za posluživanje zahtjeva. Prije plaćanja za usluge udomljavanja web stranice, potrebno je odabrati plan koji najbolje odgovara potrebama web stranice i kupiti ga. Ako se dogodi situacija da promet na web stranici raste i dolazi do potrebe za više prostora na poslužitelju, moguće je prijeći na jedan od naprednijih planova koji

nude bolje mogućnosti i svojstva [11]. Nakon odabira imena web domene stranice te usluge udomljavanje web stranice, moguće je započeti s izradom web stranice. S napretkom tehnologije te alata za izradu web stranica, usluge za udomljavanje u današnje vrijeme dolaze s integracijom gdje pružaju uslugu instalacije različitih CMS-a (eng. Content Management System) na kojima je moguće upravljati svim aspektima vlastite web stranice. Platforma WordPress, koja je korištena u ovom radu, jedna je od takvih vrsta usluga koja olakšava i ubrzava izradu web rješenja na sebi svojstven način.

2.3.4 cPanel

cPanel je jedan od najpopularnijih upravljačkih ploča koje se baziraju na operacijskom sustavu Linux, a služi za upravljanje računima koji služe za udomljavanje web aplikacije gdje je moguće upravljati svim uslugama na jednom mjestu. Nakon prve prijave na sučelje alata cPanel moguće je pristupiti podacima o upotrebi resursa kao što su korištenje CPU-a, raspoloživom prostoru za pohranu te upotrebu memorije, što može pružiti koristan način praćenja svih performansi web stranice. cPanel pruža brzu instalaciju WordPress platforme i ako se to učini, korisnik može koristiti cPanel da bi upravljao datotekama od kojih se sastoji web stranica razvijena na platformi WordPress. Isto tako je moguće upravljanje bazom podataka, imenom domene, email računom i sigurnosnom kopijom (eng. Backup) web stranice. Ako je platforma koja pruža uslugu udomljavanja web aplikacije integrirana s alatom cPanel, nakon prve instalacije se dobije email u kojem se nalazi poveznica preko koje je moguće pristupiti cPanel-u. Prema [12], postoji i drugi način pristupa gdje nije potrebno pristupati email-u i tražiti poveznicu koja vodi na cPanel upravljačku ploču, a ovisi o protokolu kojega određena web stranica koristi:

- Ako URL web stranica koristi HTTP protokol, potrebno je dodati sljedeći niz znakova na kraj domene*: 2082*
- Ako URL web stranice koristi HTTPS protokol, potrebno je dodati sljedeći niz znakova na kraj domene*: 2083*

Ako je web aplikacija udomljena na vrsti platforme koja koristi cPanel kao upravljačku ploču te ta web aplikacija koristi HTTP protokol pristup bi bio preko poveznice http://www.primjer.hr:2082, ako ta web aplikacija koristi HTTPS protokol pristupali bi preko poveznice https://www.primjer.hr:2083.

2.3.5 WordPress

WordPress je sustav za upravljanje sadržajem (eng. CMS - Content Management System) te se temelji na PHP i MySQL tehnologijama. Značajke uključuju arhitekturu dodataka (eng. plugin) i sustav predložaka (eng. template). Platforma se najviše povezuje s blog stranicama, no podržava druge vrste web sadržaja, uključujući različite oblike web stranica. Koristi ga više od 60 milijuna web stranica, uključujući 33,6% od prvih 10 milijuna web stranica do travnja 2019. [13]. Potrebno je napomenuti da postoje bitne razlike između WordPress.com te WordPress.org platforme, te razlike se nalaze u tablici 2.1, preuzete iz [14].

Tablica 2.1 Razlika između Wordpress.com i WordPress.org platforme

	WordPress.com	**WordPress.org**
Instalacija	Ništa nije potrebno instalirati, potrebna je samo prijava.	Potrebno je instalirati WordPress osobno, ručno ili preko upravljačke ploče poslužitelja.
Teme	Moguće je koristiti bilo koju temu koja je dostupna od strane WordPress.com platforme.	Moguće koristiti bilo koju temu koja je dostupna online, od bilo kojeg autora (uključujući osobnu temu).
Plugin	Nema mogućnosti dodavanje i odabira (osim kod korištenja plaćenog Premium WordPress.com modela).	Moguće koristiti bilo koji plugin dostupan online, od bilo kojeg autora (uključujući osobni plugin).
Nadogradnja	WordPress.com pruža automatske nadogradnje.	Potrebno je osobno izvršiti nadogradnju kad je ista dostupna.
Widgets	Dostupnost ovisi o odabranoj temi.	Moguće je raditi promjene na bilo kojoj temi.
Održavanje	Nije potrebno nikakvo održavanje.	Odgovornost za održavanje vlastite web stranice pripada administratoru.
Oglašavanje	Nije moguće vlastito oglašavanje. Međutim, WordPress.com ponekad vrši oglašavanje na vašoj vlastitoj web stranici.	Moguće je raditi oglašavanje u bilo kojem obliku na vlastitoj web stranici.
Vlasništvo	Iako sadržaj na samoj web stranici pripada vama kao korisniku, WordPress.com ima ovlasti da ga ukloni u bilo kojem trenutku ako smatraju da nije prikladan.	Administrator ima potpunu kontrolu nad vlastitom web stranicom, te ga nitko ne može prisiliti da istu ukloni.
Domena	Web stranica je zadana kao pod-domena unutar domene WordPress.com, ali postoji mogućnost nadogradnje na Premium paket (uz naknadu) te je na taj način moguće koristiti vlastitu domenu.	Moguće je koristiti vlastitu domenu koja je prethodno registrirana.

WordPress je najpopularniji sustav upravljanja web sadržajem koji se koristi. Da bi bio funkcionalan, WordPress mora biti instaliran na web poslužitelj te imati vlastito ime domene pomoću koje je moguće pristupiti web stranici. Korisnici platforme WordPress mogu instalirati veliki broj dostupnih tema koje omogućuju promjenu izgleda i funkcionalnosti WordPress stranice bez promjene izvornog koda. Svaka WordPress stranica zahtijeva barem jednu temu koja mora postojati, te mora biti razvijena prema WordPress standardima uz PHP, HTML i CSS programske jezike.

Za izradu novog web rješenja udruge, korištena je platforma WordPress.org, otvorenog je izvornog koda (eng. open source), besplatna je te pruža brojne mogućnosti prilagodbe web stranice po vlastitim potrebama. Sve što je potrebno za početak korištenja su registrirano ime domene i platforma koja pruža uslugu za udomljavanje web stranice. Na ovaj način je moguće imati potpunu kontrolu nad vlastitom web stranicom, a promjene je moguće raditi na efikasan način. U tablici 2.1 su prikazane razlike između platforma WordPress.com i WordPress.org, a ovdje bi valjalo naglasiti koji su razlozi odabira platforme WordPress.org, te koje su prednosti korištenja:

- Besplatan je i jednostavan za korištenje (eng. user friendly).
- Administrator posjeduje vlastitu web stranicu i sve njene podatke. Nema bojazni od isključivanja web lokacije jer se sadržaj na istoj protivi uvjetima usluge same platforme (Sve dok se ne radi nešto protivno zakonu).
- Na web stranicu je moguće dodavati besplatne, plaćene i prilagođene dodatke.
- Moguća je prilagodba dizajna web stranice po potrebi - dodavanje teme WordPress je moguće u bilo kojem trenutku razvoja.

Postoje i određeni nedostaci:

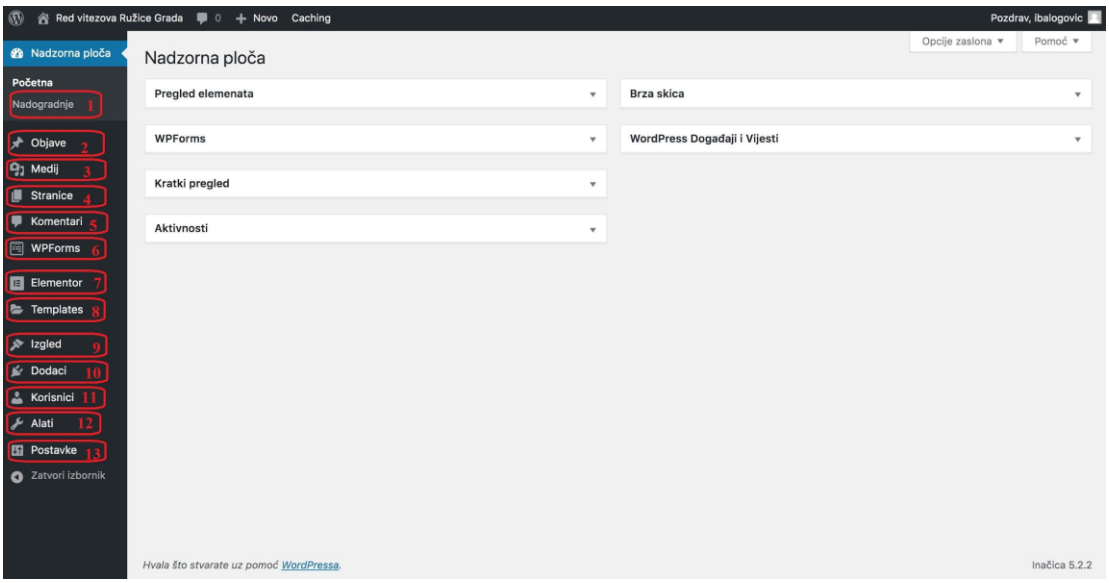
- Potrebno je osobno odabrati plan web posluživanja, a samim time treba imati na umu kakva će biti količina prometa na web stranici u smislu broja pristupa korisnika na web stranicu. Rastom prometa na web stranici postoji mogućnost potrebe za promjenom plana posluživanja.
- Administrator je odgovoran za svaku nadogradnju verzije WordPress-a te svih dodataka koje koristi.
- Administrator je odgovoran za sigurnosnu kopiju (eng. backup) cijele web lokacije, a ta radnja se ne odrađuje automatski, no postoje *plugin* rješenja koja olakšavaju proces stvaranja sigurnosne kopije [15].

U nastavku poglavlja pružen je detaljniji opis WordPress.org platforme gdje je fokus stavljen na dijelove te značajke platforme koje su korištene za izradu novog web rješenja udruge.

2.3.6 Nadzorna ploča

Nakon prijave vlastitim podacima na platformu, dolazi se do administrativnog dijela WordPress stranice. Prva stranica koja je vidljiva nakon prijave se zove Nadzorna ploča (eng. dashboard), zbirka podataka o aktivnostima i radnjama na WordPress stranici gdje se nalaze podaci te informacije o promjenama na web stranici. Također su vidljive sve nedavne promjene, odnosno izvješća o nedavnim aktivnostima, ažuriranjima sustava i obavijestima koja su vezana uz dodatke i teme koje su instalirane na web stranicu. Početna stranica nadzorne ploče se sastoji od više elemenata, *Pregled elemenata* predstavlja sažetak aktivnosti gdje je moguće vidjeti broj objava, stranica te komentara koji se nalaze na samoj web stranici. Svaka od ovih vrsta sadržaja je prikazana u obliku poveznice, kojoj je moguće pristupiti. Na dnu *widgeta* se nalazi trenutna WordPress verzija te tema koje su povezane uz samu web stranicu. *Widget Aktivnosti* predstavlja nedavne komentare sa stranice, kojima je moguće upravljati. *Brza skica* predstavlja način za brzo sastavljanje nove objave gdje je moguće napisati naslov objave, priložiti određeni medij kao sliku, napisati njen sadržaj, dodati oznake te tu objavu spremiti kao skicu ili je odmah objaviti na samoj web stranici.

Nadzorna ploča koja predstavlja administratorski dio stranice sadrži veliki broj funkcija, koje su dostupne uz pristup izborniku koji se nalazi na lijevoj strani zaslona, a te su funkcije vidljive na slici 2.6.



Slika 2.6 Prikaz prozora nadzorne ploče WordPress platforme

Na lijevoj strani zaslona je glavni navigacijski izbornik koji detaljno opisuje svaku administrativnu funkciju koju je moguće obaviti. Na slici 2.6 je prikazan je navigacijski izbornik koji se sastoji od 13 kartica (eng. tab). Kartica pod nazivom *Nadogradnje* se koristi kada je potrebno ažurirati WordPress, dodatke ili teme, a kad je dostupna nova verzija pojavit će se broj koji pokazuje koliko je ažuriranja potrebno izvršiti. Kartice pod rednim brojem 2 i 4 se koriste za objavljivanje dvaju vrsta sadržaja te su zadani načini na koje je moguće stvoriti sadržaj na web lokaciji. Kartica *Objave* je najbolja za objavu sadržaja u stilu bloga, dok je kartica *Stranice* bolja za objavu statičkog sadržaja. Kartica *Medij* služi za upravljanje slikama i drugim medijskim datotekama koje je moguće prenijeti na WordPress web lokaciju. U kartici *Komentari* je moguće upravljati komentarima posjetitelja web lokacije. Kartica pod rednim brojem 6, *WPForms* predstavlja dodatak pomoću kojeg je moguće dodati kontakt formu na samu stranicu, preko koje korisnik može kontaktirati vlasnika web lokacije. Kartica *Elementor* predstavlja *page builder* koji olakšava izradu sučelja web stranice, te pruža veliki broj različitih mogućnosti prilagodbe. Kartica pod rednim brojem 8, pod nazivom *Templates* predstavlja predložak koji definira dio web stranice generiran WordPress temom. Kao primjer je moguće navesti *header.php* koji je zadani predložak koji se koristi u većini WordPress tema. Određuje područje zaglavlja web stranice koje generira WordPress. Datoteka zaglavlja obično se učitava na svakoj stranici određene WordPress stranice, omogućujući izmjene u jednoj datoteci, koje će se primjenjivati na cijeloj web stranici [16]. Kartica *Dodaci* (eng. plugins) predstavljaju programske skripte koje dodaju različite funkcionalnosti na web stranicu. Nakon dolaska na karticu *Dodaci* moguće je vidjeti popis svih instaliranih dodataka na web lokaciji. Također je moguće aktivirati, deaktivirati ili izbrisati svaki dodatak iz ovog područja. Sljedeća kartica, *Korisnici* predstavlja dio nadzorne ploče gdje je moguće pregledati i upravljati svim računima koji su registrirani na određenoj web lokaciji, ako postoji samo jedna osoba koja ima pristup toj web lokaciji, onda ta kartica nije toliko značajna. Kartica *Alati* je korisna za mogućnost uvoza ili izvoza sadržaja (eng. Content), što je korisno ako dođe do potrebe za premještajem web lokacije. Posljednja kartica, pod nazivom *Postavke* sadrži sve postavke koje definiraju web lokaciju u cjelini. Postavke određuju kako web stranica funkcionira, moguće je konfigurirati više parametara, kao što su vremenska zona, jezik, format datuma, struktura URL poveznica, upravljati komentarima te još mnoge druge stavke.

2.3.7 WordPress tema

Tema predstavlja izgled WordPress stranice, odnosno njezino korisničko sučelje. Određena tema povlači sadržaj i podatke koji su pohranjeni od strane WordPress-a te ih prikazuje u web pregledniku, tema predstavlja skup datoteka koje su međusobno povezane da bi korisničko sučelje

bilo objedinjeno u jednu smislenu cjelinu, a te datoteke predstavljaju predloške (eng. template files). Pojedina tema može sadržavati prilagođene datoteke predložaka, slike u različitim formatima, CSS datoteke, prilagođene stranice, isto tako potrebne programske skripte (na primjer .php datoteke) [17]. Pojam *Template file* se koristi na različite načine u domeni tema:

- Datoteke koje predstavljaju predloške (eng. template files) postoje unutar same teme te izražavaju na koji je način sama web stranica prikazana.
- Predlošci stranica (eng. page templates) predstavljaju predložak koji se odnosi samo na stranice i oni određuju izgled pojedine stranice, odjeljka stranice (eng. page section) ili klasu stranica.
- Oznake predložaka (eng. template tags) su ugrađene WordPress funkcije koje je moguće koristiti unutar datoteka koje predstavljaju predloške da bi se dohvatili i prikazali podaci.
- Hijerarhija predložaka (eng. template hierarchy) je logika koju WordPress koristi da bi odlučio koje datoteke predložaka treba, ovisno o sadržaju koji se traži (što je zatražio sam korisnik).

Dvije najbitnije datoteke od kojih se tema sastoji su index.php, koja predstavlja glavnu datoteku u kontekstu predložaka te style.css, koja predstavlja stilski predložak koji kontrolira prezentaciju (dizajn i raspored) pojedinih stranica same web stranice kao cjeline.
Dakle, tema WordPress se sastoji od datoteka koje predstavljaju predloške. To su PHP datoteke koje sadrže kombinaciju HTML-a, oznaka predložaka i PHP koda. Recimo da datoteka pod nazivom "header.php" predstavlja predložak koji omogućuje stvaranje zaglavlja same stranice. Nakon što korisnik posjeti određenu stranicu, WordPress učitava predložak na osnovu zahtjeva, a vrsta sadržaja koja je prikazana od strane datoteke predloška određena je prema vrsti objave (eng. post type) koja predstavlja različiti sadržaj na stranici (bio to naslov ili pojedina informacija u podnožju stranice). Hijerarhija predložaka opisuje koju datoteku predloška WordPress učitava, a to se temelji na vrsti zahtjeva te na tome da li taj predložak uopće postoji u temi. Poslužitelj nakon toga obrađuje (eng. Parse) PHP u predlošku te vraća HTML posjetitelju WordPress stranice. Najvažnija datoteka u temi je index.php, ona je obavezna u svim temama zato što dohvaća sve predloške koji su navedeni u temi.

2.3.8 Dodaci WordPressa

Dodatak (eng. plugin) je skripta programskog koda koja proširuje osnovnu namjenu i funkcionalnost samog WordPress-a, njegova namjena je da proširi trenutnu ili doda novu

funkcionalnost na web stranicu. Dodaci WordPress su pisani u programskom jeziku PHP, te se veoma jednostavno integriraju s WordPress-om. Dodavanje novog *plugina* je moguće napraviti kroz administracijski dio platforme WordPress, no prije dodavanja je potrebno provjeriti da li je kompatibilan s trenutnom WordPress verzijom. Prednost dodataka je u tome da umjesto promjene jezgre programske podrške WordPressa-a, oni dodaju novu funkcionalnost na stranicu, koja ima vrlo učinkovitu integraciju.

2.4 Testiranje programske podrške

Tradicionalno gledajući, nastojanje da se poboljša kvaliteta određenog proizvoda usredotočeno je na kraj razvojnog ciklusa naglašavanjem otkrivanja i ispravljanja pogrešaka. Novi pristup provjere te poboljšanja kvalitete obuhvaća sve faze procesa razvoja proizvoda - od analize zahtjeva do konačne isporuke proizvoda kupcu. Svaki korak u procesu razvoja mora biti izveden na najbolji mogući način, a učinkovit proces provjere kvalitete mora se usredotočiti na sljedeće:

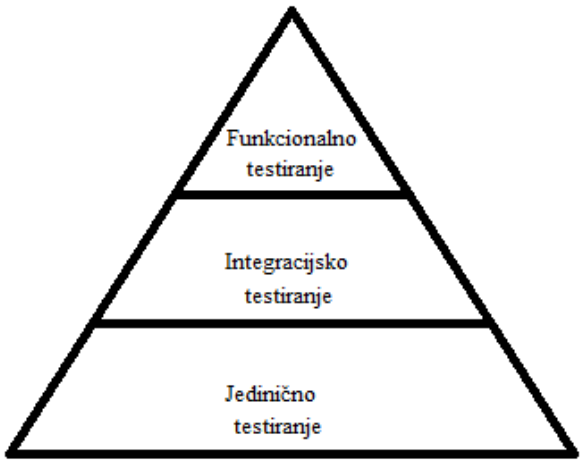
- Potrebno je obratiti pozornost na korisničke zahtjeve.
- Potrebno je neprestano poboljšavati kvalitetu proizvoda.
- Integriranje mjernih procesa s dizajnom i razvojem proizvoda.
- Razvijanje perspektive na razini sustava s naglaskom na metodologiju i procese.
- Uklanjanje viška stalnim usavršavanjem [18].

2.4.1 Kvaliteta programske podrške

Kvaliteta programske podrške predstavlja razinu prema kojoj sustav, komponenta ili proces zadovoljava potrebe odnosno očekivanja korisnika. Može se reći da je to skup funkcionalnosti i značajki programskog proizvoda koje ovise o njegovoj sposobnosti da zadovolji navedene potrebe [19]. Kvaliteta programske podrške sadrži mnogo komponenata, neke od njih su: pristupačnost, kompatibilnost, učinkovitost, funkcionalnost, sposobnost održavanja, performanse, pouzdanost, skalabilnost, sigurnost, te mogućnost testiranja. Dakle, da bi potvrdili da je neki programski proizvod kvalitetan, potrebno je procesom testiranja provjeriti da li je to stvarno tako. Prema standardu ANSI/IEEE 1059, testiranje programske podrške je proces analize programske podrške određenog proizvoda s ciljem otkrivanja razlika između trenutnih i početnih zahtjeva te da bi se procijenile njegove značajke [20].

2.4.2 Podjela testiranja programske podrške

Prije prikaza alata koji su prilagođeni za proces testiranja, potrebno je prikazati koje vrste testiranja programske podrške postoje i na kojoj razini one najbolje odgovaraju. Testna piramida (eng. testing pyramid) koju je prvi definirao Mike Cohn je model koji timovi koriste kako bi pokazali koje su tri različite vrste testova za osiguravanje kvalitete programske podrške i kako se međusobno nadopunjuju. Model testne piramide prikazan je na slici 2.6, a napravljen je prema [21].



Slika 2.6 Testna piramida

Određena web stranica se sastoji od sloja korisničkog sučelja koji sadrži gumbe (eng. button) te poveznice koje korisnici koriste da bi se navigirali po stranici. Postoji i servisni sloj (eng. service layer) koji šalje UI sloju podatke koji su mu potrebni za ažuriranje zaslona. Posljednji je sloj mozak cijelog sustava (eng. logic layer) koji je zadužen za učinkovito funkcioniranje cijelog sustava. Naravno da nisu sve aplikacije građene na ovaj način, postoje aplikacije koje nemaju korisničko sučelje, no principi testne piramide svejedno stoje. Na vrhu piramide, UI (eng. User Interface) test nazivamo testiranjem korisničkog sučelja. Ova vrsta testa prolazi kroz cijeli sustav i ponaša se slično kao što bi se ponašao korisnik kada bi koristio programski proizvod, može uključivati provjeru različitih funkcionalnosti kao što je navigacija kroz izbornik, interakcija s ikonama i gumbima, unos podataka u različite vrste forma, te bilo koje druge funkcionalnosti na web stranici. Integracijski testovi su poput UI testova, samo što ne provode testiranje putem korisničkog sučelja, nego umjesto toga idu jedan sloj ispod te izravno testiraju osnovne usluge koje čine samo korisničko sučelje. Svrha integracijskog testiranja je otkriti greške u interakciji između različitih komponenti sustava, koji se nazivaju moduli (eng. module). Modul općenito razvija jedan programski inženjer čije se razumijevanje i programska logika može razlikovati od drugih

programskih inženjera, gdje je u tom slučaju integracijsko testiranje potrebno da bi se ustanovilo da različiti programski moduli funkcioniraju zajedno, gdje moduli mogu biti stranica za prijavu (eng. login page), spremnik elektroničke pošte te neka druga funkcionalnost, od kojih svaka može biti integrirana drugačije. Posljednja razina (eng. unit tests) testiranja programske podrške je razina na kojoj se testiraju pojedine jedinice/komponente programa. Svrha je provjere da svaka komponenta programske podrške radi kako je zamišljeno, gdje komponenta predstavlja najmanji dio koji je moguće testirati, a sam proces testiranja se izvršava tijekom razvoja programske podrške. Cilj ove vrste testiranja je izolirati dio programskog koda te provjeriti njegovu ispravnost. Na kraju je potrebno napomenuti da je za potrebe testiranja u ovom radu najpogodnije testiranje korisničkog sučelja koje pruža pravu vrijednost u procjeni kvalitete web stranice zato što najbolje simulira radnje pravog korisnika te samim time pruža sigurnost da web stranica funkcionira kao jedan povezan sustav i onako kako je zamišljeno. Također je moguće otkrivati greške tijekom samog razvoja web aplikacije, te ih je moguće u realnom vremenu i popraviti

2.4.3 Testiranje korisničkog sučelja

Testovi korisničkog sučelja se izvode pomoću programske skripte koja testira određenu aplikaciju, odnosno jednu njezinu funkciju, na isti način na koji bi krajnji korisnik imao interakciju s tom komponentom sustava. Skripta vrši automatizaciju radnji kao što su klik, dodir, odabir, prijava i učini nešto bi korisnik sam učinio. Razlog zbog kojeg su UI testovi korisni je njihova mogućnost da u principu prolaze kroz svaki sloj aplikacije, što se još naziva i *end-to-end* testiranje, što znači da takva vrsta testa prolazi kroz svaki dio aplikacije, kroz korisničko sučelje, kroz sve funkcionalnosti te kroz bazu podataka. To je razlog zbog kojeg su UI testovi toliko dobri u testiranju povezanosti različitih komponenti sustava, zbog čega ih često koristimo na visokoj razini pod nazivom *Smoke tests*. Pogodni su jer jamče da na nekoj minimalnoj razini sustav funkcionira pravilno. Potrebno je definirati na koji način UI testovi funkcioniraju.

Da bi se pojedini test uspješno izvršio, potrebno je dohvatiti element kojeg je potrebno testirati, odnosno element kojem je potrebno testirati određenu funkcionalnost. Osnovna mogućnost je dohvaćanje same stranice te dohvaćanje elemenata stranice preko različitih identifikatora, koji mogu biti ID, ime pripadajuće klase, ime elementa, ime oznake, XPath i drugi. Izbor identifikatora koji predstavlja određeni element na stranici je specifičan, a ovisi o ciljanom rezultatu korisnika i strukturi internetske stranice. Okviri (eng. framework) za testiranje korisničkog sučelja rade sve te funkcije, oslanjajući se na dvije ključne tehnologije: HTML i CSS. HTML (HyperText Markup Language) je jezik kojim opisujemo izgled web stranica pomoću HTML oznaka, one u tekstualnom obliku određuju položaj i način prikazivanja elemenata web stranice. Web preglednik

interpretira HTML oznake i stvara prikaz web stranice. Recimo da je potrebno razviti dio stranice koja sadrži naslov i određeni sadržaj u obliku rečenice. To je moguće učiniti pomoću programskog koda 2.1:

```
<h1>Opis</h1>
<p>Ovo je rečenica</p>
```

Programski kod 2.1 HTML jezik

Znakovi u zagradama (h1, p) nazivaju se oznake (eng. tags), kad se oznake stavljaju oko sadržaja koji se želi opisati, taj sadržaj se u tom slučaju označava. Kod dohvaćanja elemenata treba biti oprezan, zato što se može dogoditi da je napisan UI test koji ovisi o relativnom položaju nekog elementa, te nakon promjene izgleda taj isti element više nije na istoj poziciji pa test koji je napisan više nije valjan. Preferirani način dohvaćanja elementa je dohvaćanjem jedinstvenog identifikatora, takav identifikator je dan u programskom kodu 2.2.

```
<input id="email" type="text">
```

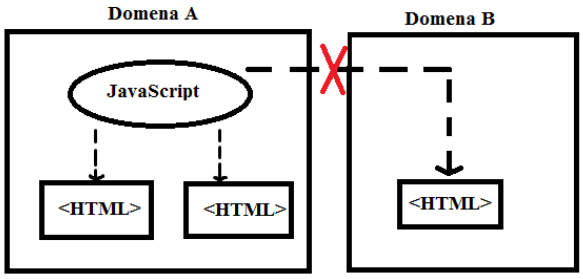
Programski kod 2.2 Primjer identifikatora

U programskom kodu 2.2, identifikator email predstavlja ID polja u koje je moguće unijeti tekst. Ta linija programskog koda može predstavljati jedno polje za unos u određenoj registracijskoj formi, gdje je pomoću ID-a pod nazivom email moguće pristupiti tom polju.

2.4.4 Selenium

Veliki broj aplikacija danas se izvodi u web-pregledniku, a učinkovitost testiranja ovakvih aplikacija varira među tvrtkama i organizacijama. U doba interaktivnih programskih procesa u kojima mnoge organizacije koriste neki oblik Agilne metodologije, testna automatizacija često postaje zahtjev za projekte razvijanja programske podrške. Automatizacija testiranja predstavlja korištenje programskih alata koji pokreću ponovljive testove da bi izvršili proces provjere ispravnosti aplikacija koje su odabrane za testiranje. Postoji mnogo prednosti za provođenje automatiziranih testova, većina ih je povezana s mogućnošću da se test ponovi te brzinom kojom se ti testovi izvršavaju. Postoji mnogo komercijalnih i *open source* alata dostupnih za provođenje automatiziranih testova, Selenium je vjerojatno jedan od najkorištenijih. Selenium je skup programskih alata koji pružaju podršku za automatizirano testiranje [22]. Služi kao alat koji oponaša interakciju korisnika s određenom web aplikacijom tako da kontrolira web preglednik i prepoznaje elemente korisničkog sučelja. Trenutno podržava većinu najpopularnijih web preglednika, a testne skripte mogu biti napisane u više različitih programskih jezika. Selenium se

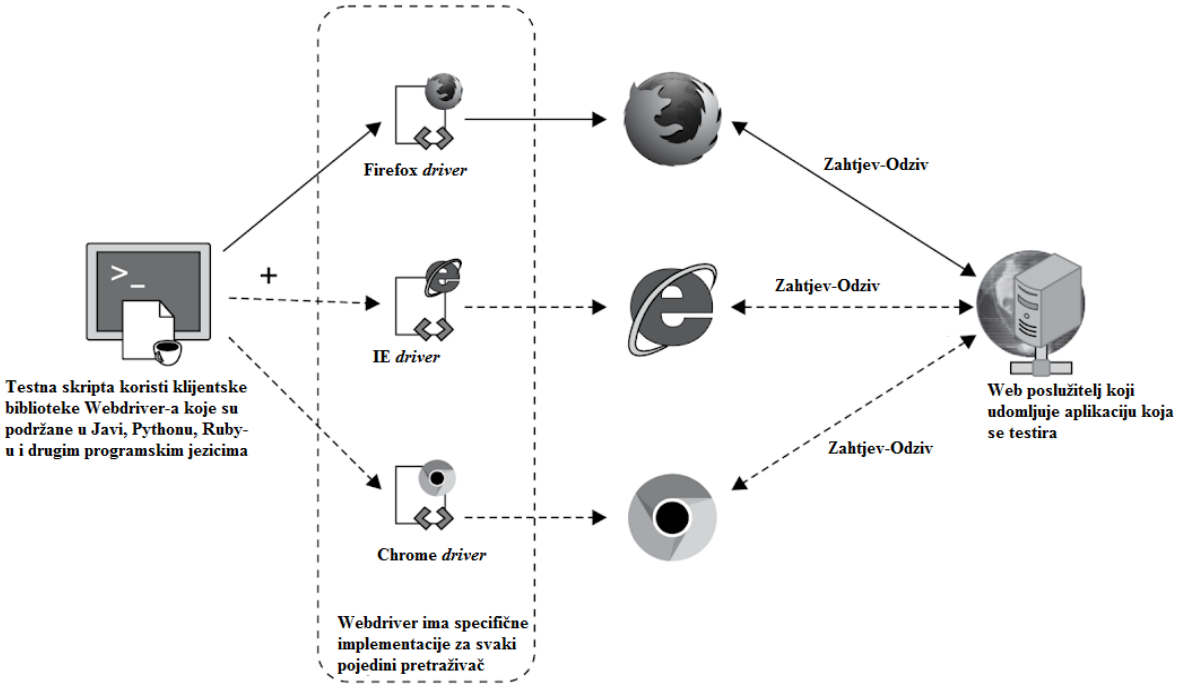
sastoji od 4 različite komponente. Pri počecima razvoja platforme Selenium, prije stvaranja komponente Selenium RC, inženjeri koji su testirali određenu web aplikaciju trebali su instalirati lokalnu kopiju skripte JavaScript (*Selenium Core*) i lokalnu kopiju poslužitelja koji sadrži web aplikaciju koju je potrebno testirati, a razlog je bio sigurnosni koncept pod nazivom *Same-origin policy* [23]. Taj koncept ne dopušta određenim skriptama da pristupe elementima druge web stranice, odnosno druge web domene. Na primjer, HTML programski kod na stranici www.google.com koristi skriptu pod nazivom "skripta.js". Gore spomenuti koncept dopušta skripti pristup stranicama unutar domene google.com kao što su google.com/email ili google.com/prijava. Međutim, ta ista skripta ne može pristupiti stranicama s drugih web mjesta zato što pripadaju drugim domenama, što je prikazano na slici 2.7, napravljenoj prema [24].



Slika 2.7 Sigurnosni koncept *Same-origin policy*

Pod sigurnosnim konceptom koji je prikazan na slici 2.7, datoteka koja sadrži JavaScript kod može pristupiti samo stranicama pod istom domenom. Komponenta Selenium RC ili jednostavnije Selenium 1, osmišljena je od strane inženjera imena Paul Hammant da bi olakšala samo testiranje i izbjegla probleme sa sigurnosnim konceptom *Same-origin policy*. Druga komponenta od koje se sastoji Selenium je Selenium Grid koji predstavlja alat koji se koristi zajedno sa alatom Selenium RC u svrhu izvođenja paralelnih testova na različitim uređajima i različitim web-preglednicima istovremeno čime se štedi na vremenu. Treća komponenta je Selenium IDE (eng. Integrated Development Environment) koji predstavlja alat koji se koristi za razvijanje testnih slučajeva i on je dodatak za web-preglednike Google Chrome i Firefox. Način na koji funkcionira je da bilježi korisničke radnje u web-pregledniku koristeći Selenium naredbe, s parametrima koji su definirani kontekstom elementa koji je namijenjen za testiranje. WebDriver je u početku nastao kao zaseban alat, kojeg je osmislio inženjer Simon Stewart 2006. Godine, a predstavljao je prvi alat za testiranje kojeg je bilo moguće koristiti na više različitih platformi i koji je mogao kontrolirati web-preglednik s razine operacijskog sustava. WebDriver se pokazao kao bolji alat od alata Selenium IDE i Selenium RC, a jedan od glavnih čimbenika je da na puno bolji, moderniji i stabilniji način

implementira automatizaciju radnji web-preglednika. Za razliku od Selenium-a RC, WebDriver kontrolira web-preglednik na način da direktno komunicira s njim, te podržava iste programske jezike koji su Java, C#, PHP, Python, Perl i Ruby. Zbog sve većih zahtjeva, 2008. godine Selenium je povezan sa projektom WebDriver, te je nazvan Selenium WebDriver, a na slici 2.8 je prikazan njegov rad [25].



Slika 2.8 Rad alata Selenium WebDriver

Kada je skripta za potrebe testiranja izvršena, dogode se sljedeći koraci:

- Za svaku naredbu je kreiran HTTP zahtjev koji je poslan upravljačkom programu (eng. driver) web-preglednika.
- Upravljački program koristi HTTP poslužitelj da bi dohvatio HTTP zahtjev.
- HTTP poslužitelj utvrđuje korake koji su potrebni za implementaciju Selenium naredbe.
- Koraci implementacije se izvršavaju u web-pregledniku.
- Nakon izvršavanja naredbe, odziv je poslan HTTP poslužitelju.
- HTTP poslužitelj šalje odziv nazad automatizacijskoj skripti.

Način na koji korisnik to može učiniti je koristeći klijentske biblioteke koje WebDriver pruža u različitim programskim jezicima, kao što su Java, Ruby, Python, Perl, PHP i .NET. Koristeći ugrađene klijentske biblioteke, korisnik može prizvati (eng. invoke) WebDriver implementaciju koja je specifična za svaki pojedini web-preglednik, odnosno Firefox Driver, IE Driver, Chrome *driver*, da bi bilo moguće komunicirati s aplikacijom koju je potrebno testirati. Te implementacije

WebDriver-a rade nativno s web-preglednikom i izvršavaju naredbe izvan web-preglednika, te simuliraju radnje koje bi radio korisnik. Nakon izvršavanja skripte, WebDriver šalje rezultate testa nazad testnoj skripti na analizu programeru.

2.4.5 JavaScript

JavaScript je skriptni programski jezik, koji se inicijalno izvršavao u web-pregledniku na strani korisnika, oznaka JS se često odnosi na naziv JavaScript, koji je uz HTML i CSS jedna od glavnih tehnologija na kojima se temelji *World Wide Web*. JavaScript omogućuje funkcioniranje velikog broja interaktivnih web stranica i suštinski je dio web aplikacija, a veliki broj web-preglednika sadrži JavaScript mehanizam (eng. engine) koji izvršava skripte. JavaScript je inicijalno implementiran na klijentskoj strani web-preglednika, no u današnje vrijeme su JavaScript mehanizmi ugrađeni u druge vrste programske podrške, uključujući i web poslužitelje i baze podataka koje su na poslužiteljskoj strani [26]. JavaScript je potrebno spomenuti jer je on jedan od programskih jezika koji je kompatibilan s alatom Selenium WebDriver, te će biti korišten za pisanje programskih skripti koje će biti korištene za automatizirano testiranje novog web rješenja udruge.

3 KORISNIČKO TESTIRANJE

Cilj poglavlja je prikazati izgled i funkcionalnost trenutnog web rješenja udruge "Red vitezova Ružice grada". Nadalje, nakon prikaza trenutnog web rješenja dan je na uvid proces korisničkog testiranja (eng. User testing), odnosno na koji način su prikupljeni podaci korisničkog zapažanja nakon interakcije s trenutnom web stranicom udruge. Na kraju poglavlja su prikazani podaci korisničkog testiranja gdje je na grafikonu moguće vidjeti korisnička zapažanja za svaku pojedinu stranicu trenutnog web rješenja.

3.1 Trenutno web rješenje udruge

Razlog zbog kojeg je došlo do ideje o izradi novog web rješenja udruge je izgled trenutnog web rješenja koji nije u skladu s današnjim standardima i nije u skladu s očekivanjima koja prosječni korisnik ima kada posjećuje određenu web stranicu, odnosno moglo bi se reći da nije u skladu s korisničkim iskustvom kakvo je danas standardno. Na slici 3.1 je prikazan izgled trenutnog web rješenja, odnosno izgled početne stranice, kojem je moguće pristupiti preko sljedeće poveznice: http://www.vitezovi-ruzicegrada.hr/.

Slika 3.1 Prikaz prozora početne stranice trenutnog web rješenja udruge

Nakon učitavanja trenutnog web rješenja, moguće je primijetiti da se prije prikaza web stranice na zaslonu ekrana pojavi grb udruge. Nakon svakog učitavanja web stranice, pojavljuje se logo udruge na koji je potrebno kliknuti da bi se učitala sama stranica. Nakon učitavanja web stranice,

pojavljuje se početna stranica, koja je vidljiva na slici 3.1. Prva stvar koja je vidljiva, odnosno najistaknutija je datum koji vjerojatno predstavlja održavanje sljedećeg turnira što pokazuje da stranica nije održavana s obzirom na to da je prikazana 2015. godina kao godina sljedećeg turnira. Na početnoj stranici se, osim datuma sljedećeg turnira nalazi Facebook ikona, koja bi trebala voditi na Facebook stranicu udruge, no to se ne dogodi nakon pokušaja klika na istu. Nadalje, nakon klika na ikonu "Aktualno", otvara se stranica koja je vidljiva na slici 3.2.

Slika 3.2 Prikaz prozora stranice "Aktualno"

Još jedan pokazatelj slabog održavanja web stranice je i datum turnira koji se nalazi na stranici "Aktualno", potrebno je napomenuti da je nejasno na kojem se dijelu web stranice korisnik nalazi nakon klika na ikonu zato što se URL struktura web stranice ne mijenja bez obzira na pozicioniranje na stranici. Na slici 3.3 se nalazi stranica "Arhiva" na kojoj je jedini sadržaj popis godina gdje nije jasno što on točno predstavlja. Jedina pretpostavka koju je moguće donijeti da je to arhiva određenog sadržaja kroz godine od početka postojanja udruge, no ni to nije ažurirano s obzirom to da je zadnja godina koja stoji u popisu 2014.

Slika 3.3 Prikaz prozora stranice "Arhiva"

Zaključak je da je međusobna interakcija različitih dijelova na web stranici dosta zbunjujuća, a glavni razlog je nepostojanje URL strukture poveznica, s kojom bi bilo puno lakše zaključiti gdje pojedina ikona vodi. Također, stranica "Galerija" sadrži popis prethodnih događaja na kojima je udruga bila, gdje je moguće pogledati fotografije sa pojedinih događaja. Na slici 3.4 je vidljiv izgled te stranice.

Slika 3.4 Prikaz prozora stranice "Galerija"

Na slici 3.4 je vidljivo da na stranici galerija ikona zelene boje prekriva tekst koji predstavlja naziv određene galerije fotografija s turnira na kojem je udruga bila sudionik. Također, nakon otvaranja pojedine galerije nema jasnog pokazatelja koji bi ukazao o kojem se turniru radi, a sama navigacija

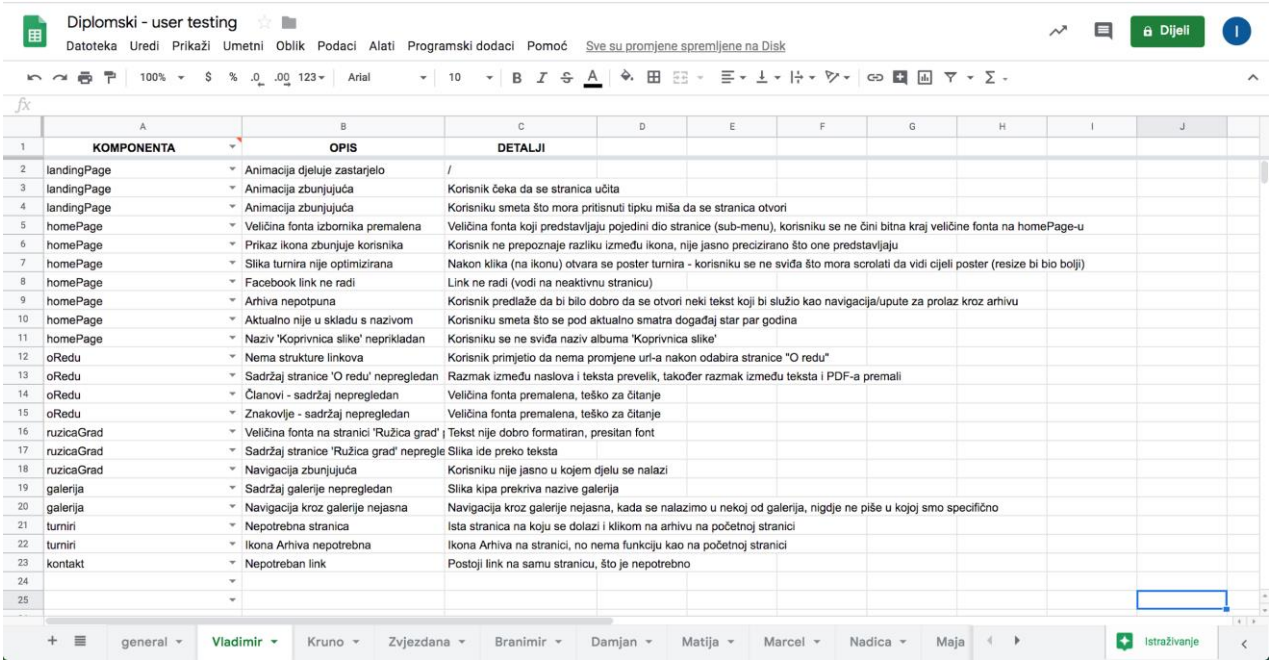
unutar određene galerije pomalo je zbunjujuća jer nedostaju tipke za navigaciju te za izlaz iz galerije. Na slici 3.5 se nalazi stranica "Kontakt" koja sadrži kontakt podatke odgovornih osoba u udruzi, te adresu same udruge.

Slika 3.5 Prikaz prozora stranice "Kontakt"

Dodatak koji je moguće dodati na stranicu za kontakt podatke bi bio prikaz adrese udruge pomoću alata Google, tako da je adresu moguće provjeriti u alatu *Google maps*, unutar same web stranice, što olakšava lociranje sjedišta udruge, bez odlaženja sa same stranice.
Cilj ovog poglavlja je bio prikazati izgled trenutnog web rješenja udruge sa što manje primjedbi i osobnih zapažanja, što je cilj sljedećeg poglavlja gdje će korisnici odrediti značajke razvoja novog web rješenja za udrugu.

3.2 Korisničko testiranje web rješenja udruge

Korisničko testiranje trenutne web stranice udruge "Red vitezova Ružice grada" provedeno je u seriji intervjua s korisnicima. Testiranje je provedeno na testnoj skupini određenog broja ljudi koja se sastoji od nekoliko inženjera iz tvrtke "FIVE", a ostatak čine ljudi koji nemaju iskustvo razvijanja ili testiranja programske podrške, s ciljem dobivanja što boljeg uvida u stanje trenutnog web rješenja udruge, te na koje stvari je potrebno obratiti pozornost kod izrade novog web rješenja. Na slici 3.6 je prikazan izgled datoteke u kojoj su spremljeni podaci korisničkog testiranja.



Diplomski - user testing

Datoteka Uredi Prikaži Umetni Oblik Podaci Alati Programski dodaci Pomoć Sve su promjene spremljene na Disk

Dijeli

	KOMPONENTA	OPIS	DETALJI
2	landingPage	Animacija djeluje zastarjelo	/
3	landingPage	Animacija zbunjujuća	Korisnik čeka da se stranica učita
4	landingPage	Animacija zbunjujuća	Korisniku smeta što mora pritisnuti tipku miša da se stranica otvori
5	homePage	Veličina fonta izbornika premalena	Veličina fonta koji predstavljaju pojedini dio stranice (sub-menu), korisniku se ne čini bitna kraj veličine fonta na homePage-u
6	homePage	Prikaz ikona zbunjuje korisnika	Korisnik ne prepoznaje razliku između ikona, nije jasno precizirano što one predstavljaju
7	homePage	Slika turnira nije optimizirana	Nakon klika (na ikonu) otvara se poster turnira - korisniku se ne sviđa što mora scrolati da vidi cijeli poster (resize bi bio bolji)
8	homePage	Facebook link ne radi	Link ne radi (vodi na neaktivnu stranicu)
9	homePage	Arhiva nepotpuna	Korisnik predlaže da bi bilo dobro da se otvori neki tekst koji bi služio kao navigacija/upute za prolaz kroz arhivu
10	homePage	Aktualno nije u skladu s nazivom	Korisniku smeta što se pod aktualno smatra događaj star par godina
11	homePage	Naziv 'Koprivnica slike' neprikladan	Korisniku se ne sviđa naziv albuma 'Koprivnica slike'
12	oRedu	Nema strukture linkova	Korisnik primjetio da nema promjene url-a nakon odabira stranice "O redu"
13	oRedu	Sadržaj stranice 'O redu' nepregledan	Razmak između naslova i teksta prevelik, također razmak između teksta i PDF-a premali
14	oRedu	Članovi - sadržaj nepregledan	Veličina fonta premalena, teško za čitanje
15	oRedu	Znakovlje - sadržaj nepregledan	Veličina fonta premalena, teško za čitanje
16	ruzicaGrad	Veličina fonta na stranici 'Ružica grad'	Tekst nije dobro formatiran, presitan font
17	ruzicaGrad	Sadržaj stranice 'Ružica grad' nepregle	Slika ide preko teksta
18	ruzicaGrad	Navigacija zbunjujuća	Korisniku nije jasno u kojem djelu se nalazi
19	galerija	Sadržaj galerije nepregledan	Slika kipa prekriva nazive galerija
20	galerija	Navigacija kroz galerije nejasna	Navigacija kroz galerije nejasna, kada se nalazimo u nekoj od galerija, nigdje ne piše u kojoj smo specifično
21	turniri	Nepotrebna stranica	Ista stranica na koju se dolazi i klikom na arhivu na početnoj stranici
22	turniri	Ikona Arhiva nepotrebna	Ikona Arhiva na stranici, no nema funkciju kao na početnoj stranici
23	kontakt	Nepotreban link	Postoji link na samu stranicu, što je nepotrebno

general | Vladimir | Kruno | Zvjezdana | Branimir | Damjan | Matija | Marcel | Nadica | Maja | Istraživanje

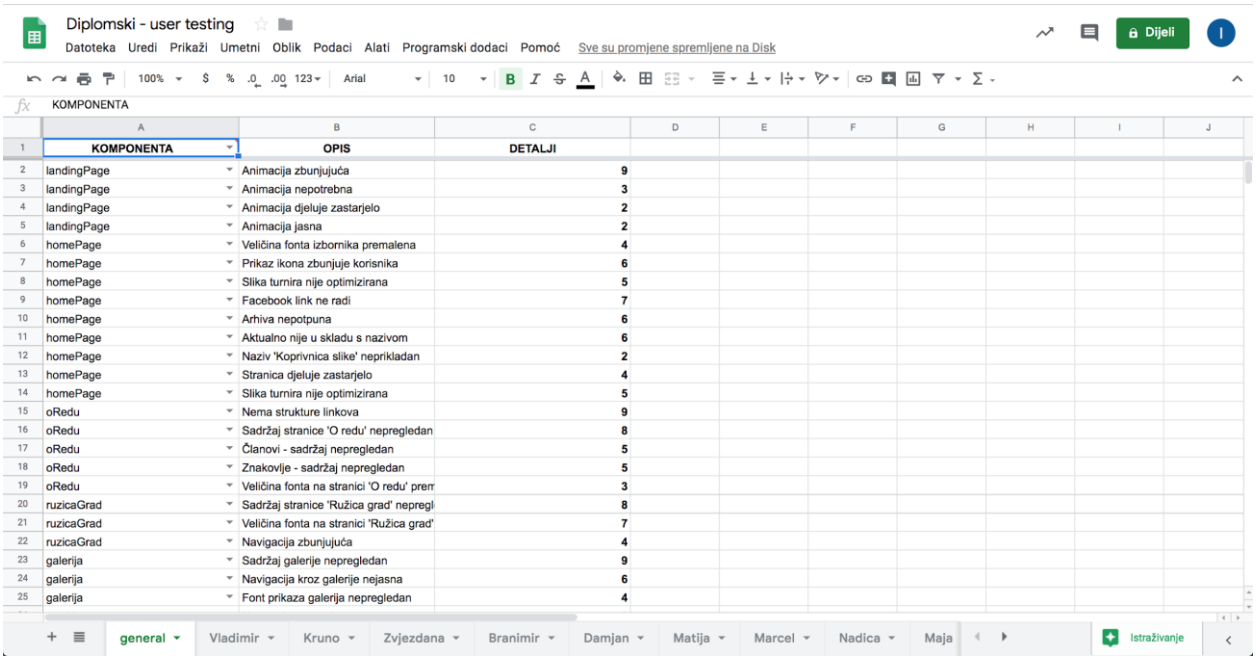
Slika 3.6 Primjer podataka korisničkog testiranja jednog korisnika

Na slici 3.6 se nalaze podaci jednog od korisnika koji je sudjelovao u korisničkom testiranju. Kao što je prikazano na slici, svaka kartica (eng. tab) s imenom predstavlja pojedinačnog korisnika, a kartica pod nazivom *General* predstavlja skup podataka svih korisnika i služi za stvaranje prikaza podataka u obliku grafikona. Podaci koji se nalaze u tablici prikazuju korisničko iskustvo (eng. user experience) tijekom korištenja web stranice udruge koje je zabilježeno u audio formatu, te obliku bilježaka. Interakcija koju je korisnik imao sa pojedinim komponentama na web stranici udruge prikazana je u tablici u obliku korisnikovih zapažanja gdje stavka “Komponenta” označava pojedini dio web rješenja na koji se odnosi korisnikovo zapažanje, stavka “Opis” označava zapažanje korisnika na pojedinu komponentu web stranice, a stavka “Detalji” opisuje to zapažanje na detaljniji način. Komponente koje se nalaze u stupcu A predstavljaju pojedine dijelove i stranice na web rješenju udruge:

- *landingPage* - predstavlja animaciju u obliku grba udruge, koja se pojavljuje kod svakog učitavanja web stranice.
- *homePage* - predstavlja početnu stranicu trenutnog web rješenja.
- *oRedu* - predstavlja stranicu “O redu”, gdje se nalaze informacije o udruzi.
- *ruzicaGrad* - predstavlja stranicu “Ružica grad”, na kojoj se nalaze informacije o istoimenoj srednjovjekovnoj utvrdi.
- galerija - predstavlja skup fotogalerija turnira na kojima je udruga sudjelovala.

- turniri - predstavlja stranicu na kojoj se nalazi opis pojedinih turnira na kojima je udruga sudjelovala, koja još sadrži određenu galeriju fotografija.
- kontakt - predstavlja stranicu na kojoj se nalaze informacije za kontakt udruge, te adresa.

Navedena kartica "General" predstavlja sumu svih zapažanja korisnika po pojedinačnim komponentama trenutnog web rješenja udruge, što je vidljivo na slici 3.7.


Diplomski - user testing

Datoteka Uredi Prikaži Umetni Oblik Podaci Alati Programski dodaci Pomoć Sve su promjene spremljene na Disk

Dijeli

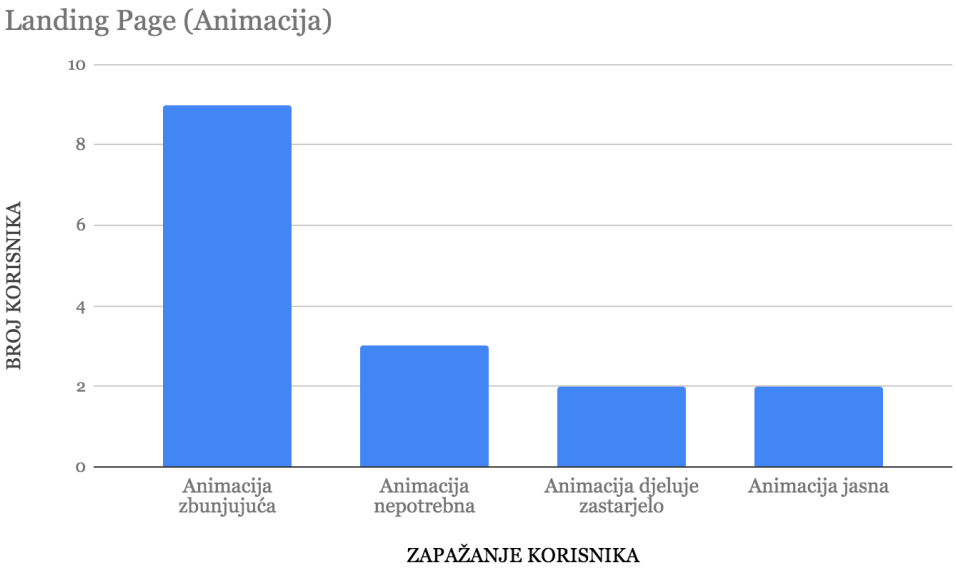
KOMPONENTA

	A	B	C
1	KOMPONENTA	OPIS	DETALJI
2	landingPage	Animacija zbunjujuća	9
3	landingPage	Animacija nepotrebna	3
4	landingPage	Animacija djeluje zastarjelo	2
5	landingPage	Animacija jasna	2
6	homePage	Veličina fonta izbornika premalena	4
7	homePage	Prikaz ikona zbunjuje korisnika	6
8	homePage	Slika turnira nije optimizirana	5
9	homePage	Facebook link ne radi	7
10	homePage	Arhiva nepotpuna	6
11	homePage	Aktualno nije u skladu s nazivom	6
12	homePage	Naziv 'Koprivnica slike' neprikladan	2
13	homePage	Stranica djeluje zastarjelo	4
14	homePage	Slika turnira nije optimizirana	5
15	oRedu	Nema strukture linkova	9
16	oRedu	Sadržaj stranice 'O redu' nepregledan	8
17	oRedu	Članovi - sadržaj nepregledan	5
18	oRedu	Znakovlje - sadržaj nepregledan	5
19	oRedu	Veličina fonta na stranici 'O redu' prem	3
20	ruzicaGrad	Sadržaj stranice 'Ružica grad' nepregl	8
21	ruzicaGrad	Veličina fonta na stranici 'Ružica grad'	7
22	ruzicaGrad	Navigacija zbunjujuća	4
23	galerija	Sadržaj galerije nepregledan	9
24	galerija	Navigacija kroz galerije nejasna	6
25	galerija	Font prikaza galerija nepregledan	4

general Vladimir Kruno Zvjezdana Branimir Damjan Matija Marcel Nadica Maja Istraživanje

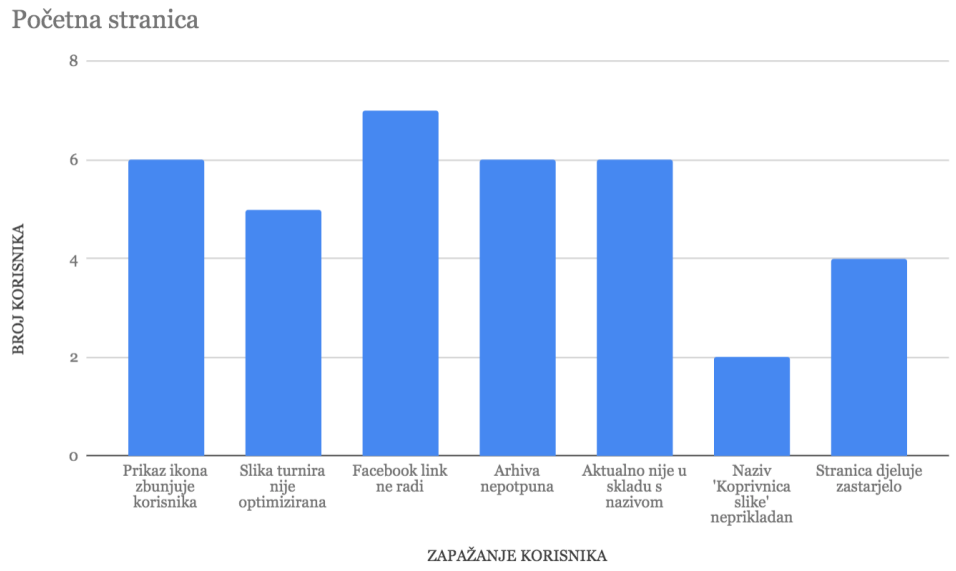
Slika 3.7 Suma podataka korisničkog testiranja svih korisnika

Podaci koji prikazuju korisničko iskustvo prikazani su na slici 3.7. Kao što je prikazano na slici, podaci su podijeljeni na tri različite komponente. Stavka pod nazivom komponenta označava pojedinu stranicu na web rješenju, dok stavka pod nazivom opis predstavlja zapažanje korisnika. Devet je korisnika primijetilo da im je animacija koja se pojavljuje kod ponovnog učitavanja web stranice zbunjujuća. U sljedećem dijelu poglavlja prikazani su podaci korisničkog testiranja za svaki pojedini dio web rješenja, počevši s animacijom koja se pojavljuje kod učitavanja web stranice. Potrebno je napomenuti da "ZAPAŽANJE KORISNIKA" predstavlja korisničko zapažanje o određenoj funkcionalnosti na web stranici udruge, a "BROJ KORISNIKA" označava koliki je broj korisnika imao specifično zapažanje. Na slici je vidljivo da su korisnici za *landingPage*, odnosno animaciju koja se pojavljuje kod učitavanja web stranice imali 4 različita zapažanja, što je na slici 3.8 i prikazano.



Slika 3.8 Prikaz podataka korisničkog testiranja za animaciju

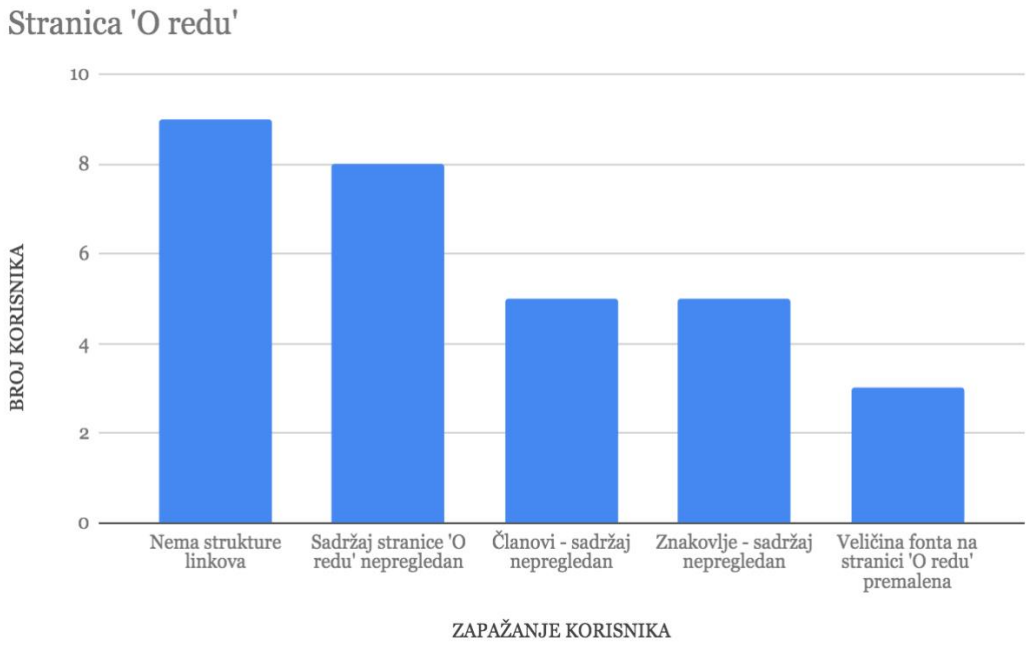
Iz podataka je očito da je animacija koja se pojavljuje kod učitavanja web rješenja nepotrebna, te će biti uklonjena kod izrade novog web rješenja udruge s obzirom na to da je devet korisnika spomenulo da im je animacija zbunjujuća. Potrebno je napomenuti da je dvoje korisnika spomenulo da im je animacija jasna, što ne mora značiti da smatraju da je potrebna na novom web rješenju udruge. Na sljedećoj slici se nalaze podaci koji prikazuju korisnička zapažanja na početnoj stranici te interakcije s elementima na istoj, što slika 3.9 prikazuje.



Slika 3.9 Prikaz podataka korisničkog testiranja za početnu stranicu

Početna stranica na trenutnom web rješenju predstavlja stranicu koja se otvara nakon zatvaranja animacije, što se dogodi nakon aktivnosti korisnika, odnosno pritiska tipke miša na prostoru ekrana web stranice. Stavka "Prikaz ikona zbunjuje korisnika" predstavlja ikone koje se nalaze na početnoj stranici (Arhiva, Aktualno i Facebook ikona) gdje nije intuitivno što točno one predstavljaju. Sljedeća stavka se odnosi na poster turnira koji se otvori nakon pritiska tipke miša na središnji dio stranice koja predstavlja turnir, fotografija nije optimizirana te nakon otvaranja nije moguće vidjeti cijeli poster. Također je sedam korisnika primijetilo da nakon pritiska tipke miša na ikonu društvene mreže Facebook dolazi do pogreške, odnosno poveznica ne vodi na stranicu udruge na društvenoj mreži Facebook. Stavka "Arhiva nepotpuna" se odnosi na zapažanje korisnika da nakon odlaska u arhivu nije jasno što godine predstavljaju, odnosno bilo bi dobro da postoji više informacija ili barem neka naznaka što bi točno trebale predstavljati. Sljedeća stavka se odnosi na ikonu "Aktualno" gdje nakon dolaska na stranicu koja predstavlja aktualne događaje stoji 2014. godina što daje dojam da web stranica nije redovito održavana. Sljedeća stavka se odnosi na neprikladan naziv galerije turnira u Koprivnici, što dvoje korisnika smatra neprikladnim. Zadnja se stavka u grafičkom prikazu podataka odnosi na općeniti izgled te dojam trenutnog web rješenja udruge, gdje je četvero korisnika primijetilo da web stranica izgleda zastarjelo.

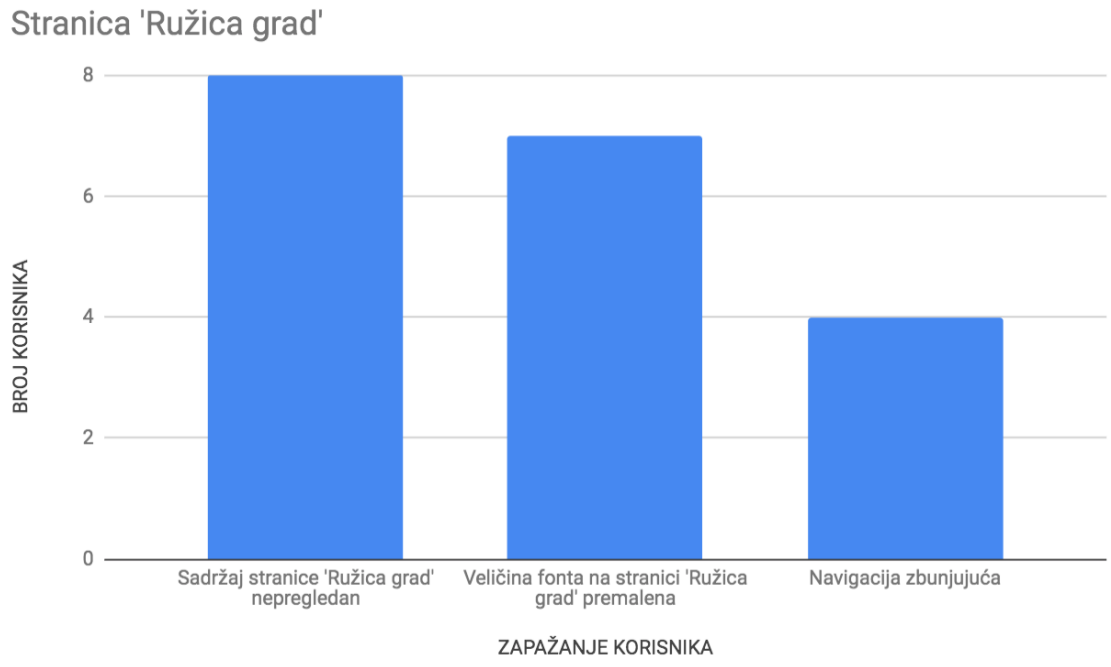
Na slici 3.10 se nalaze podaci korisničkog testiranja stranice "Ružica grad", odnosno zapažanja koja su korisnici imali nakon dolaska na istu. Kao što je vidljivo na grafičkom prikazu, zapažanja koja su korisnici imali na spomenutoj stranici se mogu podijeliti u pet različitih stavki.



Slika 3.10 Prikaz podataka korisničkog testiranja za stranicu "O redu"

Prva stavka koja se odnosi na URL strukturu označava zapažanje korisnika da se nakon promjene pojedine stranice ne mijenja URL struktura, što je primijetilo devedeset posto korisnika. Sljedeća stavka se preklapa sa zadnjom stavkom u grafičkom prikazu korisničkog testiranja, a odnose se na prikaz teksta te sadržaja na stranici "O redu" gdje je korisnicima veličina fonta premalena pa je otežano čitanje teksta, a statut udruge koji je prikazan kao PDF datoteka na dnu stranice stoji preblizu samome tekstu. Na stranici je moguće otvoriti posebnu stranicu koja sadrži popis članova udruge, gdje je korisnicima također veličina fonta premalena i to uzrokuje poteškoće kod pregleda. Uz mogućnost pristupa popisu članova reda, moguće je pristupiti stranici gdje je prikazan izgled znakovlja udruge (grb i zastava udruge) na koje korisnici nisu imali primjedbe, no na toj stranici je veličina fonta također premalena i proizlazi da se kroz cijelo web rješenja taj problem pojavljuje gdje je veličina fonta premalena za čitanje pa bi isto trebalo promijeniti kod izrade novog web rješenja.

Na slici 3.11 se nalaze podaci korisničkog testiranja stranice "Ružica grad", odnosno zapažanja koja su korisnici imali nakon dolaska na istu. Kao što je vidljivo na grafičkom prikazu, zapažanja koja su korisnici imali na spomenutoj stranici se mogu podijeliti na tri različite stavke.

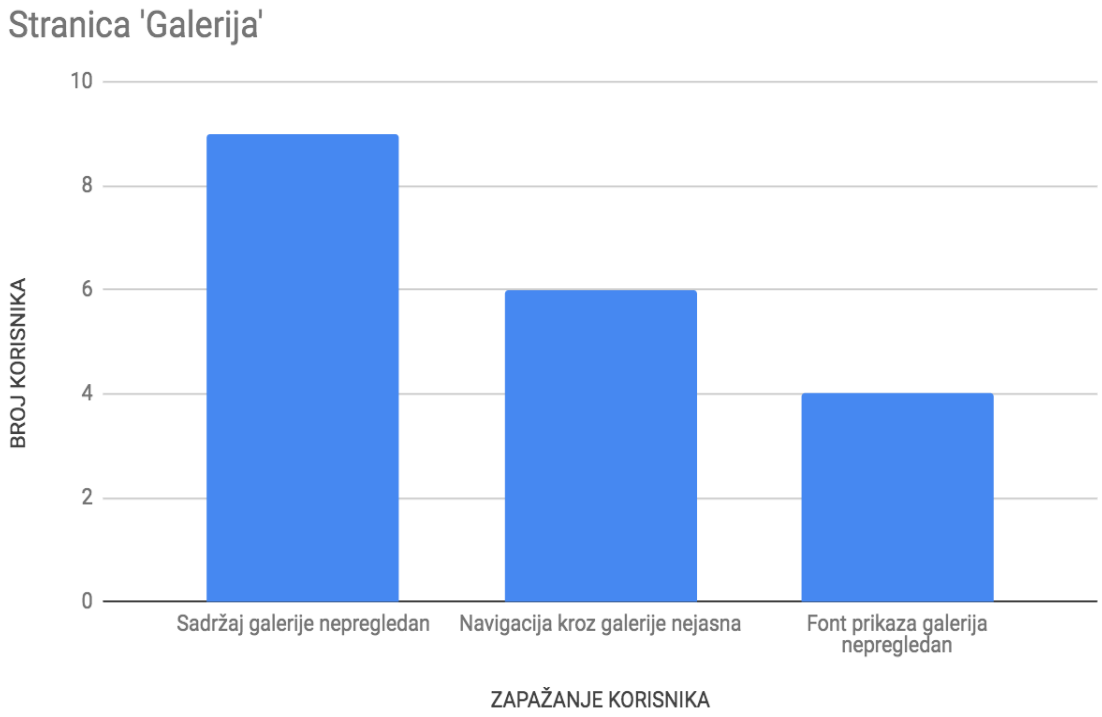


Slika 3.11 Prikaz podataka korisničkog testiranja za stranicu "Ružica grad"

Kao i u prethodnim stranicama, korisnicima je font teksta premalen. Nadalje, na stranici se nalaze različite slike koje prekrivaju tekst te je iz tog razloga sadržaj također nepregledan, gdje se tekst proteže kroz četiri različite stranice gdje uz slike navigacijske tipke prekrivaju tekst kojeg je iz tog

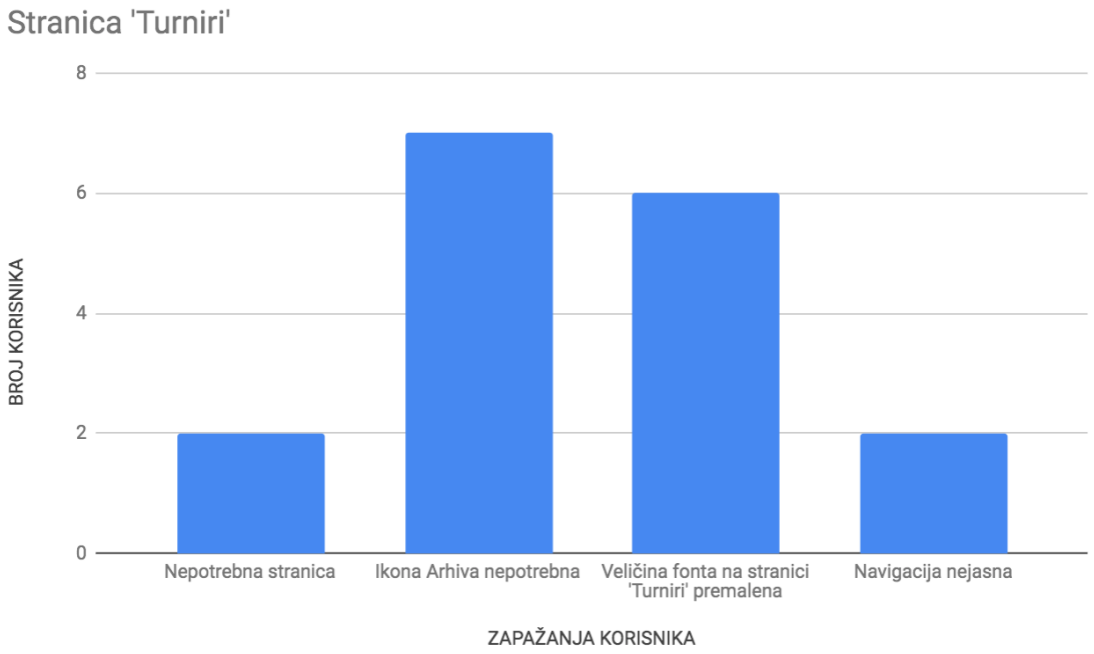
razloga teško čitati. Četvero korisnika smatra da je navigacija kroz stranicu zbunjujuća, gdje se referiraju na navigacijsku tipku koja služi za navigaciju.

Na sljedećoj se stranici nalazi slika 3.12 gdje su prikazani podaci korisničkog testiranja stranice "Galerija", odnosno zapažanja koja su korisnici imali nakon dolaska na istu. Kao što je vidljivo na grafičkom prikazu, zapažanja koja su korisnici imali na spomenutoj stranici se mogu podijeliti na tri različite stavke. Prva se stavka u grafičkom prikazu odnosi na nepregledni sadržaj, odnosno na sliku kipa koja prekriva nazive galerija gdje se ne vidi dio naziva galerije. Šestero korisnika smatra da je navigacija kroz galerije nejasna, što se odnosi na navigaciju kroz popis galerija, te navigaciju kroz određenu galeriju gdje nedostaju strelice za navigaciju i gumb za izlazak iz galerije. Zadnja se stavka u grafičkom prikazu odnosi na veličinu fonta koja označava naziv pojedine galerije, gdje bi naziv pojedine galerije trebao biti bolje naznačen.



Slika 3.12 Prikaz podataka korisničkog testiranja za stranicu "Galerija"

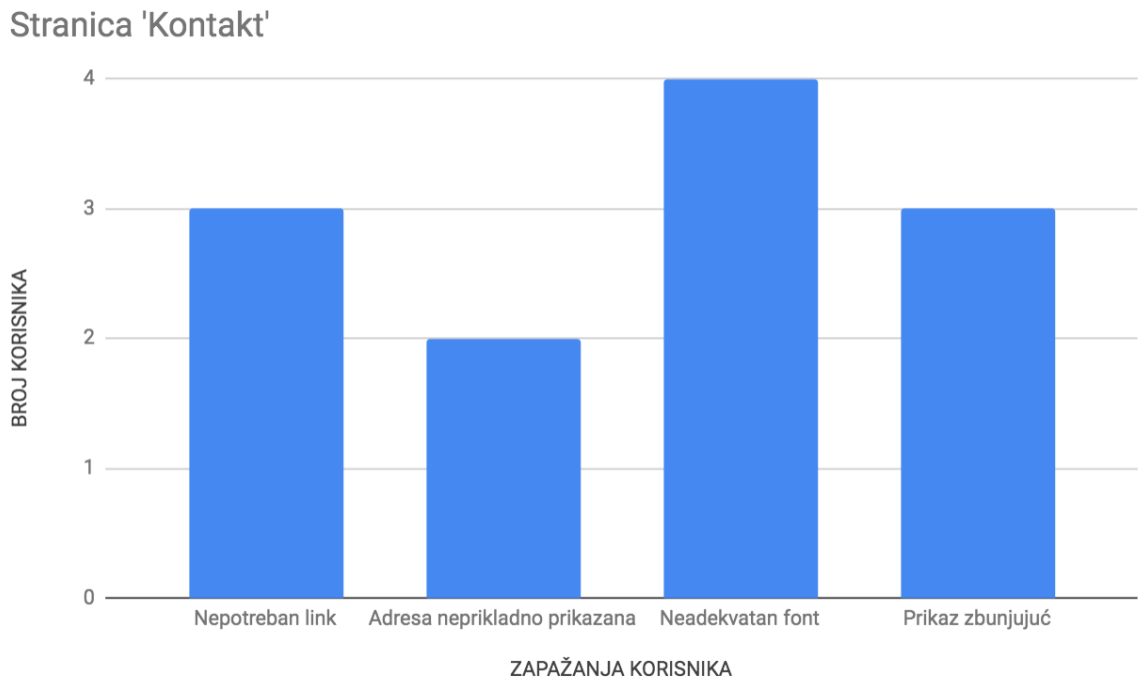
Na slici 3.13 se nalaze podaci korisničkog testiranja stranice "Turniri", odnosno zapažanja koja su korisnici imali nakon dolaska na istu. Kao što je vidljivo na grafičkom prikazu, zapažanja koja su korisnici imali na spomenutoj stranici se mogu podijeliti na četiri različite stavke.



Slika 3.13 Prikaz podataka korisničkog testiranja za stranicu “Turniri”

Prva stavka na slici 3.13 odnosi se na to da je na stranicu Arhiva moguće doći odabirom elementa koji predstavlja ime te stranice u izborniku ili klikom na ikonu Aktualno, koja se nalazi na početnoj stranici. Na stranici Turniri se također nalazi ikona Arhiva, koja nema nikakvu funkciju, a isto tako se nalazi na naslovnoj stranici, što sve zajedno može zbuniti korisnika. Veličina fonta je kao i na ostatku web rješenja korisnicima premalena. Navigacija kroz stranicu je prema mišljenju dvaju korisnika nejasna, a odnosi se na različite stranice koje je moguće otvoriti, gdje nije jasno čemu točno pripadaju, isto tako nema mogućnosti povratka na početnu stranicu, odnosno na stranicu turniri, jedina mogućnost je odabir stranice u meniju.

Na slici 3.14 se nalaze podaci korisničkog testiranja stranice “Kontakt”, odnosno zapažanja koja su korisnici imali nakon dolaska na istu. Kao što je vidljivo na grafičkom prikazu, zapažanja koja su korisnici imali na spomenutoj stranici se mogu podijeliti na četiri različite stavke.



Slika 3.14 Prikaz podataka korisničkog testiranja za stranicu “Kontakt”

Prva se stavka u grafičkom prikazu odnosi na URL poveznicu same web stranice udruge koji se nalazi na stranici za kontakt, što troje korisnika smatra da je nepotrebno. Dvoje korisnika smatra da je adresa udruge neprikladno prikazana te da bi trebala biti prikazana pomoću alata *Google maps*. Nadalje, četvero korisnika smatra da su kontakt podaci prikazani fontom koji nije adekvatan za tu vrstu podataka, odnosno da bi trebali biti bolje istaknuti, na što se također odnosi i zadnja stavka gdje troje korisnika smatra da je prikaz zbunjujuć, odnosno nema opisa koji podatak što označava.

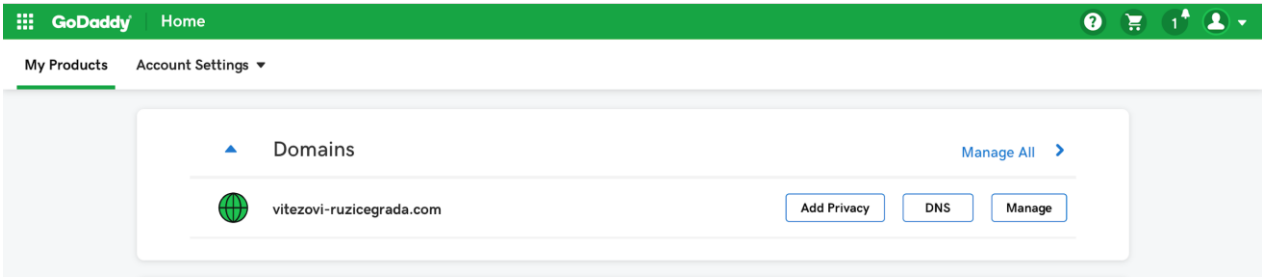
Nakon prikaza te analize podataka korisničkog testiranja potrebno je te podatke iskoristiti za što bolji i učinkovitiji razvoj novog web rješenja za udrugu gdje je potrebno obratiti posebnu pozornost na korisničko iskustvo što je i bio cilj provođenja korisničkog testiranja.

4 PROGRAMSKA PODRŠKA WEB APLIKACIJE

Cilj poglavlja je opisati postupak izrade novog web rješenja udruge, te proći kroz alate i platforme koji su korišteni za izradu. Nakon što su u drugom poglavlju definirani pojmovi internet domene, poslužitelja te WordPress platforme, odnosno svega što je potrebno za razvoj novog web rješenja udruge "Red vitezova Ružice grada", potrebno je prikazati na kojoj platformi je registrirano ime domene web stranice, gdje je udomljena te na koji je način razvijena pomoću platforme WordPress.

4.1 Registracija imena domene

Da bi korisnici mogli pristupiti određenoj web stranici, ona mora sadržavati ime domene, preko koje korisnik pristupa stranici. Ime domene mora biti jedinstveno, te ga je potrebno registrirati od strane tvrtke koja pruža usluge registracije domene. Nova web lokacija udruge registrirana je kao http://vitezovi-ruzicegrada.com/, a za to je korištena platforma GoDaddy koja pruža usluge registracije imena domena web stranica te usluge web hostinga. Za početak, da bi bilo moguće registrirati domenu, potrebno je izraditi novi račun te se registrirati na https://www.godaddy.com. Nakon dolaska na stranicu, te prve prijave na osobni račun potrebno je u tražilicu upisati željeno ime domene te provjeriti dostupnost iste, odnosno da li je trenutno u upotrebi. Ako je ime željene domene dostupno, moguće je dovršiti kupnju. Nakon prolaska kroz cijeli proces registracije domene, potrebno je potvrditi kupovinu domene, a nakon potvrde transakcije, dobije se email s podacima za prijavu. Nakon kupnje, domena je vidljiva vlastitom računu, kako je i prikazano na slici 4.1.



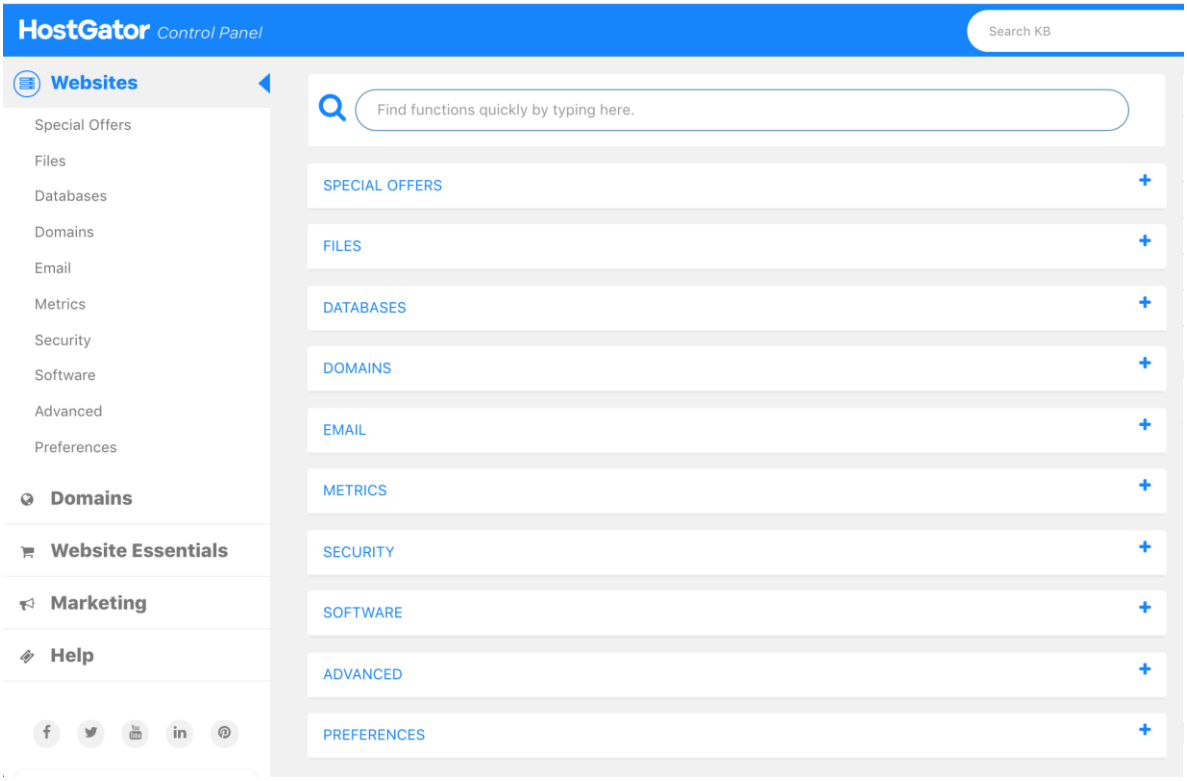
Slika 4.1 Domena web stranice udruge prikazana na platformi GoDaddy

4.2 Udomljavanje web stranice

Nakon registracije imena domene web stranice, potrebno je odabrati platformu koja nudi usluge udomljavanja web stranice, odnosno organizaciju koja nudi prostor na poslužitelju ne bi li određena web stranica bila dostupna korisnicima na internetu. Za potrebe posluživanje web stranice udruge "Red vitezova Ružice grada" odabrana je platforma HostGator (https://HostGator.com), koji nudi usluge posluživanja. Jedan od glavnih razloga odabira ove platforme za uslugu udomljavanja web stranice je i taj što nudi brzu instalaciju CMS-a kao što je WordPress, koji je moguće instalirati preko kontrolne ploče cPanel. Nakon dolaska na stranicu, te odabira usluge web hostinga, potrebno je odabrati plan koji odgovara potrebama, izbor je vidljiv na slici 4.2.

Slika 4.2 Odabir plana za udomljavanje web aplikacije na platformi HostGator

Hatchling plan bi trebao biti dovoljan za vrstu web stranice koja predstavlja udrugu, nakon odabira plana otvara se stranica na kojoj je treba upisati sve podatke koji su potrebni da bi se obavila kupnja. Na početku je moguće i registrirati domenu, što je još jedna od usluga koju nudi HostGator, no u ovom slučaju to nije potrebno jer je domena registrirana kod drugog pružatelja te vrste usluge. Nakon upisivanja svih podataka potrebno je potvrditi kupnju. Nakon dovršetka kupnje, poslani su i podaci za registraciju na cPanel, koji predstavlja kontrolnu ploču pružatelja usluge posluživanja. Izgled same kontrolne ploče vidljiv je na slici 4.3.



Slika 4.3 Kontrolna ploča HostGator-a (cPanel)

Kao što je prikazano na slici, ponuđeno je mnogo opcija za upravljanje sadržajem te postavkama web stranice koja je udomljena na platformi HostGator, kartice s lijeve strane predstavljaju značajke korisničke ploče cPanel-a, a iste se mogu naći i s desne strane, kao što je vidljivo na slici. Zato što će većina praktičnog dijela rada na web stranici za udrugu bit će obavljeno putem CMS-a WordPress platforme, sam cPanel neće biti puno korišten osim pristupa određenim skriptama programskog koda po potrebi, nadzora upotrebe resursa od strane web aplikacije, a za objašnjenje pojedinih dijelova cPanel značajki postavljena je referenca (poveznica) za dokumentaciju svih značajki cPanel-a koja se nalazi u literaturi kao [27]. Na skroz desnoj strani kontrolne ploče se nalaze informacije o cPanel računu kao što su ime trenutnog cPanel računa, trenutna glavna domena povezana s računom, osobni direktorij (eng. home directory) koji predstavlja apsolutnu putanju do osobnog direktorija računa na poslužitelju, IP adresa zadnje prijave na račun (eng. last login IP address) odnosno IP adresa uređaja osobe koja se zadnja prijavila na račun, dodatne informacije o poslužitelju, te još mnogo dodatnih informacija za koje je također postavljena referenca na dokumentaciju, te se nalazi u literaturi kao [28].

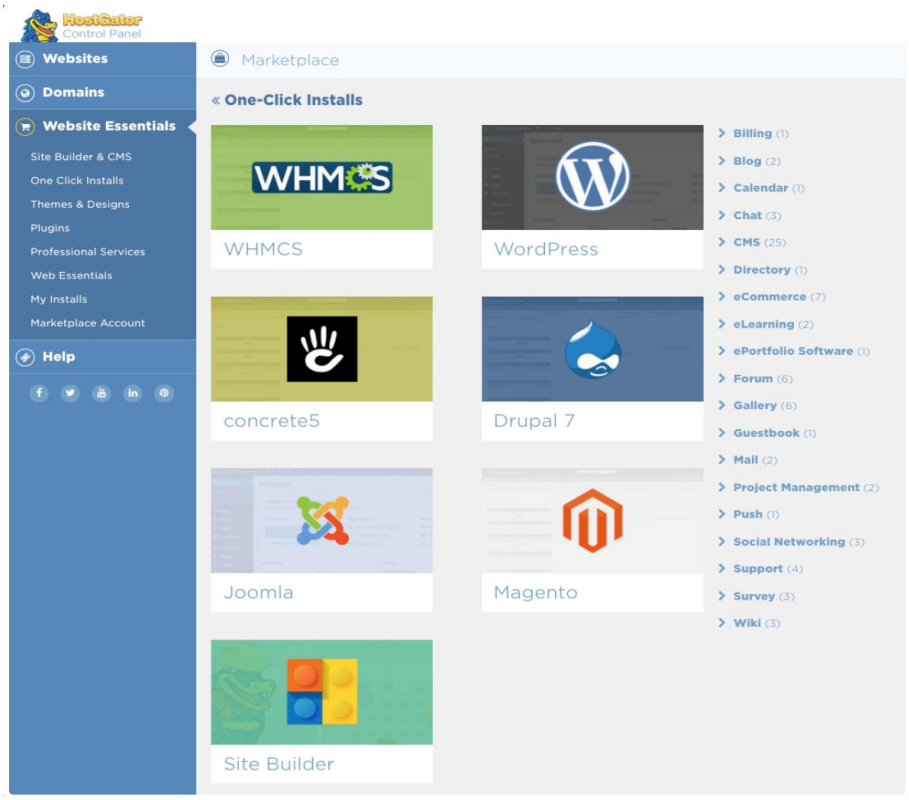
4.3 Povezivanje imena domene s web poslužiteljem

Ako dođe do situacije da je naziv web domene zauzet kod jednog pružatelja usluge, a usluga udomljavanja web aplikacije kod drugog pružatelja usluge postoji još jedan korak koji je potrebno učiniti ne bi li web stranica bila aktivna i spremna za postavljanje sadržaja. Potrebno je povezati ime web domene s imenom poslužitelja i tako da se prvo potrebno prijaviti na cPanel. Nakon toga je potrebno pronaći stavke pod nazivom Primary Nameserver i Secondary Nameserver od kojih svaka ima svoj jedinstveni naziv, obično su formata ns1.primjer.com i ns2.primjer.com. Primary Nameserver kontrolira datoteku koja sadrži administrativne informacije o samoj domeni web stranice, kao što je IP adresa domene te tko je odgovoran za administraciju te domene, a sve te informacije dobije iz lokalnih datoteka. Secondary Nameserver sadrži kopije glavne datoteke koje su samo za čitanje, a sve to dobije od prvog. Prema [29], glavna prednost koju pruža Secondary Nameserver je redundancija u slučaju da se glavni DNS server 'sruši', isto tako mogu pomoći kod distribucije zahtjeva koji dolaze prvom Nameserver-u pa da ne dođe do preopterećenja što može rezultirati s uskraćivanjem usluge korisniku. Nakon što su poznata imena servera, potrebno se prijaviti na kontrolnu ploču platforme na kojoj je registrirano ime domene web stranice, odnosno na platformu GoDaddy. Nakon prijave na GoDaddy, potrebno je otići na vlastiti račun i odabrati opciju DNS Management, te nakon učitavanja odabrati mogućnost korištenja prilagođenih poslužitelja imena. Nakon upisivanja potrebnih podataka u polja koja su predviđena za to, potrebno je završiti proces sa spremanjem novih podataka te pričekati proces promjene koji traje ne više od 24 sata. Nakon što je proces završen moguće je početi s izradom web stranice.

4.4 Model WordPress web aplikacije

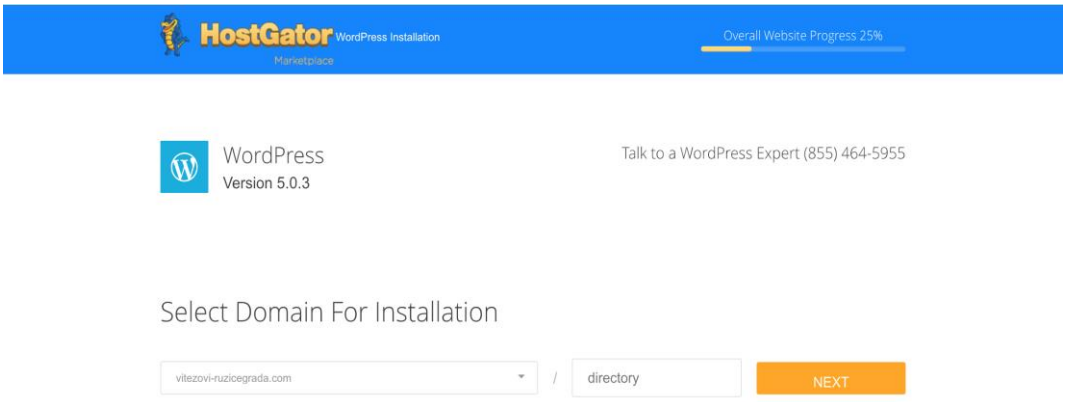
4.4.1 Instalacija i konfiguracija

Instalacija WordPress-a je izvršena putem upravljačke ploče cPanel koji omogućuje prikladno upravljanje svim uslugama na jednom mjestu, osnovno je sučelje koje se koristi za komunikaciju s uslugom web posluživanja, gdje jedna od usluga može biti instalacija Wordpress CMS-a, nakon čega se sve potrebne datoteke nalaze ondje. Wordpress predstavlja CMS (eng. *Control Management System*), odnosno sustav za upravljanje sadržajima čija je uloga olakšati i organizirati procese vezane uz kreiranje i prijenos sadržaja pojedine informacije. Općenito, CMS olakšava manipuliranje i promjenu datoteka koje služe za izradu web stranice. Nakon prijave na cPanel, na slici 4.4 je vidljivo da postoje brojni drugi sustavi za upravljanje sadržajem koje je moguće instalirati, no potrebno se navigirati do WordPress forme koja služi za instalaciju.



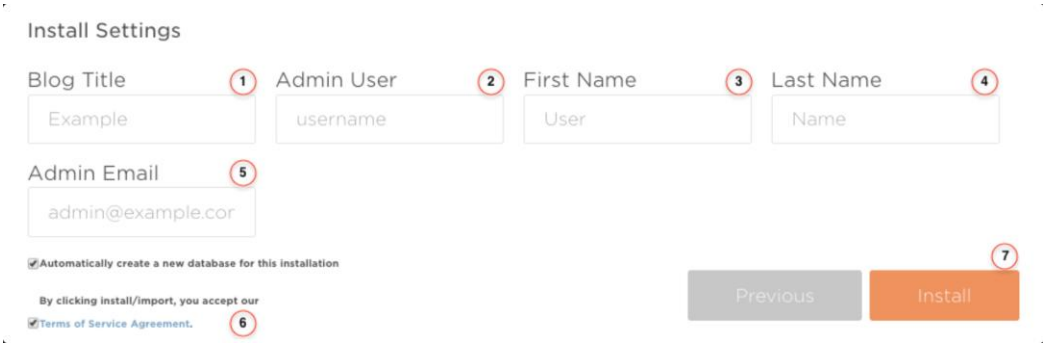
Slika 4.4 Ponuđene platforme za izradu web rješenja

Nakon odabira Wordpress CMS-a koji će biti korišten za izradu web rješenja potrebno je odabrati ime domene na koju na koju bi trebao biti instaliran sam Wordpress i nakon toga nastaviti s instalacijom, što je i prikazano na slici 4.5.



Slika 4.5 Odabir imena web domene na platformi HostGator

Sljedeći korak zahtjeva upisivanje dodatnih podataka potrebnih za završetak instalacije, detalji forme se nalaze na slici 4.6.



Slika 4.6 Postupak instalacije WordPress platforme preko alata cPanel

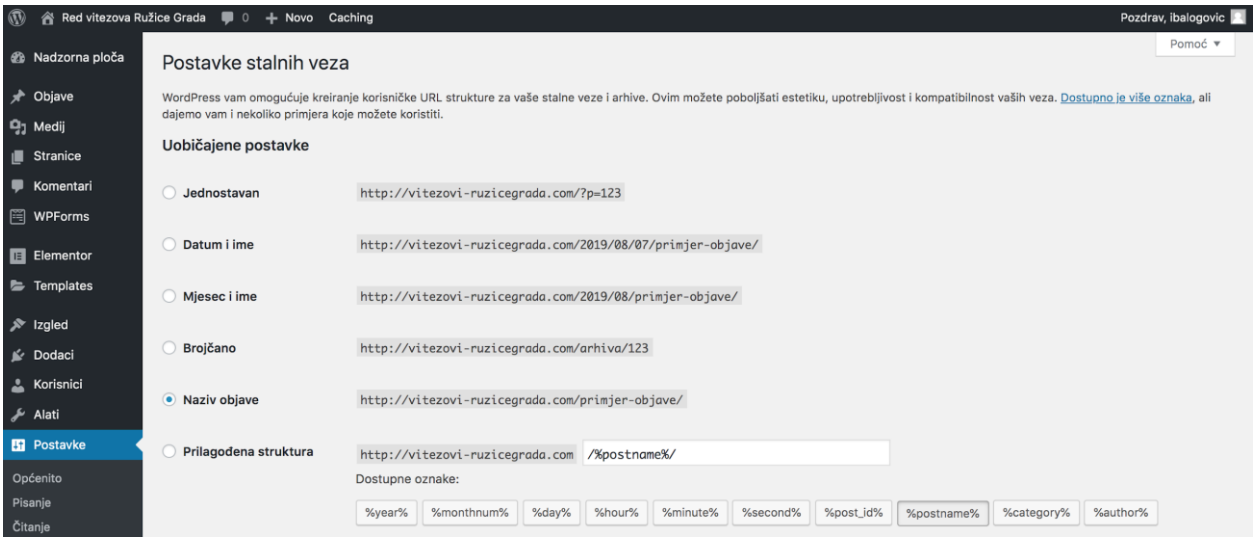
- *Blog Title* predstavlja naziv web stranice koji se nalazi u zaglavlju stranice, kasnije je taj naziv moguće promijeniti.
- *Admin User* predstavlja željeno korisničko ime.
- *First Name* i *Last Name* predstavlja ime i prezime administratora.
- *Admin Email* predstavlja email adresu koja služi za prijavu na WordPress platformu nakon instalacije.

Nakon upisivanja svih podataka, potrebno je označiti okvire za odabir (eng. c*heckbox*) te odabrati dugme za pokretanje instalacije. Sama instalacija vremenski traje najviše sat vremena, što je moguće pratiti preko trake napretka (eng. p*rogress bar*) same instalacije. Kada je instalacija završena moguće je pristupiti podacima za prijavu i obaviti prijavu na platformu WordPress.

4.4.2 Razvoj novog web rješenja udruge

Početak izrade web stranice putem platforme WordPress počinje prijavom na platformu. Sama prijava se vrši upisivanjem URL-a web stranice te dodatka /wp-admin. Dakle, nakon upisivanja poveznice http://vitezovi-ruzicegrada.com/wp-admin u tražilicu web-preglednika otvara se forma za prijavu. Nakon prijave učitava se nadzorna ploča koja predstavlja administrativni dio WordPress stranice. Prije početka izrade web stranice potrebno je konfigurirati različite funkcionalnosti u sustavu nadzorne ploče kako bi se riješili nepotrebnih dodataka koji su integrirani nakon prve instalacije, također je potrebno prilagoditi stranicu zahtjevima poput SEO (eng. *Search Engine Optimization*) optimizacije, koja se sastoji od niza aktivnosti koje su usmjerene prema podizanju posjećenosti web stranice. No prije toga je potrebno provjeriti što sve dolazi instalirano uz WordPress platformu, a nije potrebno. Treba imati na umu da svaki dadatak (eng. *Plugin*) predstavlja skupinu funkcija koje mogu proširiti funkcionalnost ili dodati nove

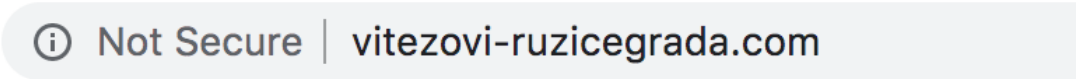
značajke na web stranicu, no isto tako mogu predstavljati dodatno opterećenje te je iz tog razloga potrebno ukloniti bilo koju wordpress temu ili dodatak koji nisu aktivni. Prvi razlog za uklanjanje nepotrebnih dodataka je responzivnost i brzina učitavanja web stranice, a drugi razlog je sigurnost. Sa stajališta sigurnosti problem koji se nameće je sljedeći, što više dodataka postoji to je više različitih funkcionalnosti na stranici koje je moguće zloupotrijebiti. Za uklanjanje svih nepotrebnih dodataka, potrebno je odabrati karticu Dodaci gdje se nalaze svi dodaci koji su instalirani, nakon toga je potrebno označiti okvire za odabir pojedinog dodatka, te nakon toga odabrati grupnu radnju Deaktiviraj, te ih nakon same deaktivacije trajno obrisati. Nakon uklanjanja nepotrebnih dodataka, potrebno je konfigurirati trajne veze (eng. *permalinks*) koje su optimizirane za tražilice, što ima veze sa pojmom SEO, a to znači da taj specifičan URL ostaje isti bez obzira na sadržaj te specifične stranice. URL struktura je jedan od dodataka u odnosu na staro web rješenje udruge, koje nije imalo URL strukture bez obzira na kojoj se stranici web rješenja korisnik nalazio. Konfiguracija izgleda URL-a odrađuje se tako da se u glavnom izborniku potrebno navigirati na postavke te odabrati opciju Stalne veze. Isto je vidljivo na slici 4.7.



Slika 4.7 Prikaz prozora s postavkama stalnih veza

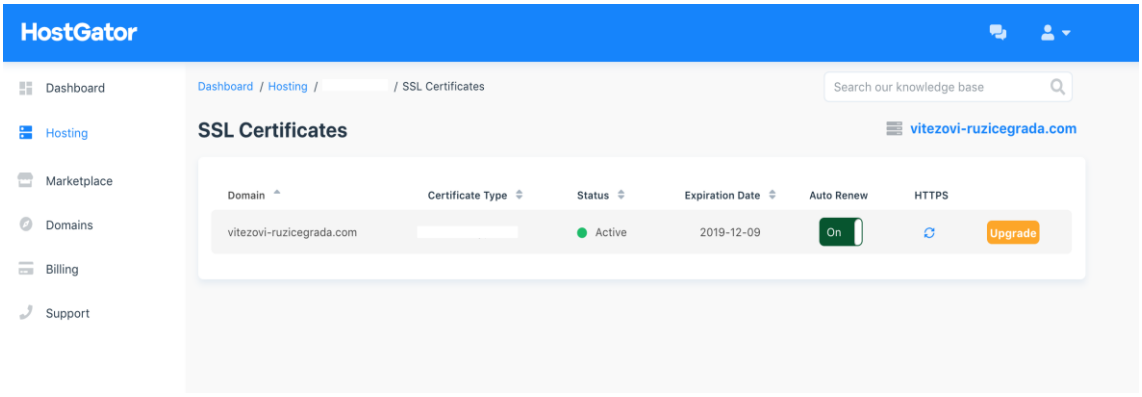
Moguće je koristiti i prilagođenu strukturu linkova, no u ovom slučaju korištena je struktura linkova s nazivom objave. Statična poveznica predstavlja strukturu URL poveznica pojedinačnih objava na web stranici. *Permalink* predstavlja URL poveznicu koja se koristi za povezivanje na određenu stranicu unutar web rješenja, a prednost je u tome da daje mogućnost jednostavnog upućivanja korisnika na dio web rješenja koji ga zanima. URL svake pojedine stranice mora biti statičan i nikada se ne bi trebao promijeniti, zato naziv trajna veza. Sljedeća stvar na koju je potrebno obratiti pozornost je sigurnost za posjetitelje web stranice. Nakon otvaranja web stranice

udruge u Google Chrome web-pregledniku te pogleda na URL tražilicu, vidljivo je sigurnosno upozorenje, koje se nalazi na slici 4.8.



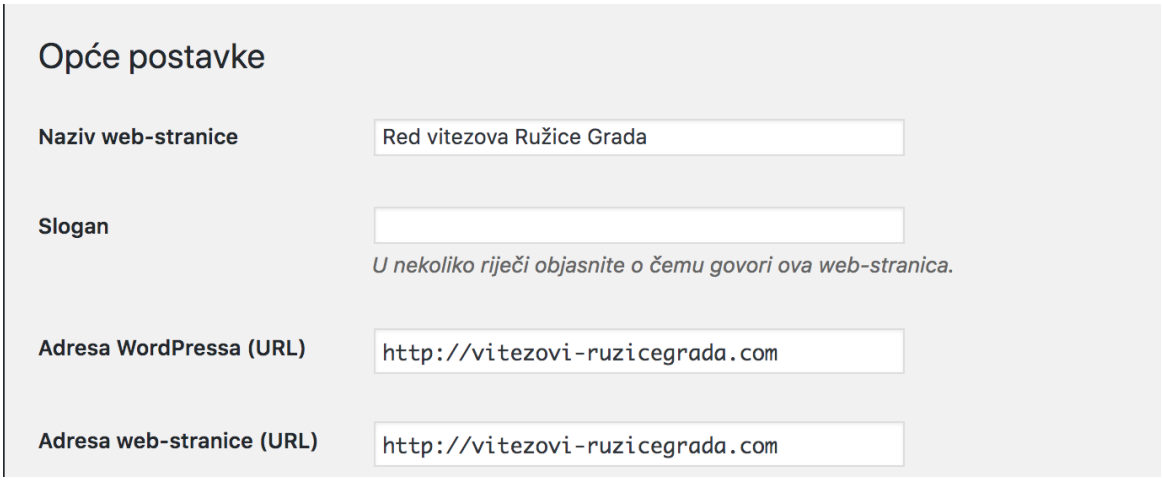
Slika 4.8 Sigurnosno upozorenje u web tražilici Google Chrome

Ako web stranica nema HTTPS (eng. HyperText Transfer Protocol Secure) certifikat, posjetitelji će dobiti upozorenje da web stranica nije sigurna, a Google Chrome, Mozilla i ostali web-preglednici će označiti svaku web stranicu kao nesigurnu (eng. Not secure) ako nakon provjere zaključe da HTTPS certifikat nedostaje. HTTPS je sigurna verzija HTTP protokola, koji služi za slanje podataka između web stranice i web preglednika, razlika je u tome da HTTPS koristi određenu kripciju (eng. encrypt), odnosno šifriranje s namjerom da se poveća sigurnost samog podatkovnog prijenosa. To je naročito važno ako korisnici prenose osjetljive podatke. S HTTPS certifikatom se povećava povjerenje posjetitelja web stranice i povećava se sigurnost korisničkih podataka tijekom interakcije na web stranici [30]. HTTPS koristi protokol da bi kriptirao komunikaciju, protokol se naziva TLS (eng. Transport Layer Security), koji također nosi i ime SSL (eng. Secure Sockets Layer). Potrebno je provjeriti da li je certifikat aktivan za web stranicu udruge, gdje se prvo treba prijaviti na cPanel HostGator-a te navigirati na ikonu *SSL Certificate,* nakon klika na ikonu, otvara se korisnički portal HostGator-a gdje je moguće vidjeti informacije o vlastitom računu, a jedna od njih predstavlja mogućnost pristupa informacijama koje pokazuju da li je HTTPS certifikat aktivan, što se nakon dolaska na stranicu potvrdilo točno, jedan od razloga odabira HostGator-a kao pružatelja usluge udomljavanja web aplikacije je taj da on nudi ugrađen HTTPS certifikat, što je vidljivo na slici 4.9.



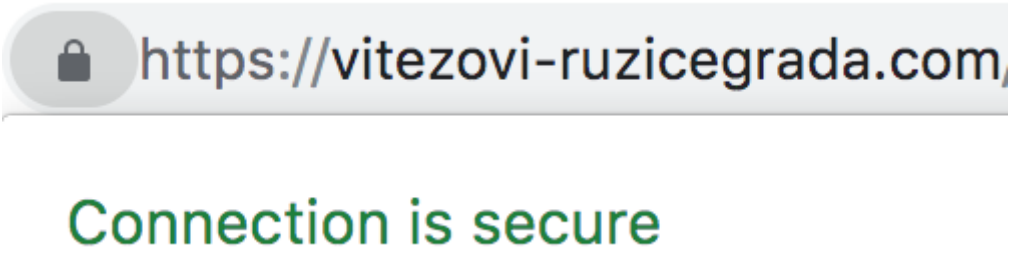
Slika 4.9 SSL certifikat prikazan na platformi HostGator

Nakon toga je potrebno promijeniti URL poveznicu na nadzornoj ploči WordPress-a, te se nakon prijave na istu potrebno navigirati u Opće postavke te ručno napraviti promjenu iz HTTP u HTTPS, na slici 4.10 je vidljiv izgled URL-a prije promjene.



Slika 4.10 Prikaz prozora s općim postavkama

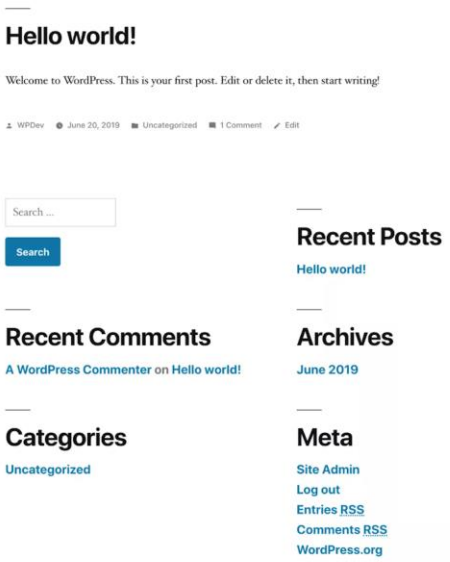
Na slici 4.11 vidljiv je izgled URL-a i nova obavijest koja ukazuje da je korisnikova veza sigurna.



Slika 4.11 Sigurnosna poruka u web tražilici Google Chrome

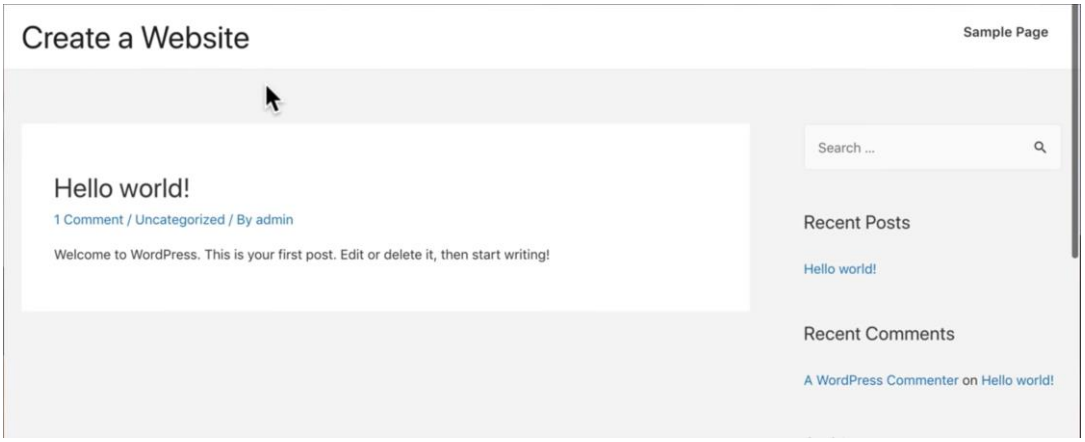
4.4.3 Promjena teme

Potrebno je napomenuti da nakon prve instalacije, te konfiguracije WordPress-a postoji zadani (eng. default) izgled web stranice, koji nije prilagođen, a vidljiv je na slici 4.12. Izgled web stranice koji se nalazi na slici definiran je WordPress temom. To je stilski izgled određene web stranice, koji služi kao okvir sadržaja. Na desnoj strani vidljivi su određeni naslovi i poveznice, što predstavlja bočnu traku (eng. Sidebar) web stranice. Stavke unutar bočne trake nazivaju se *widgeti*, dok je glavni srednji dio stranice područje sadržaja [31].



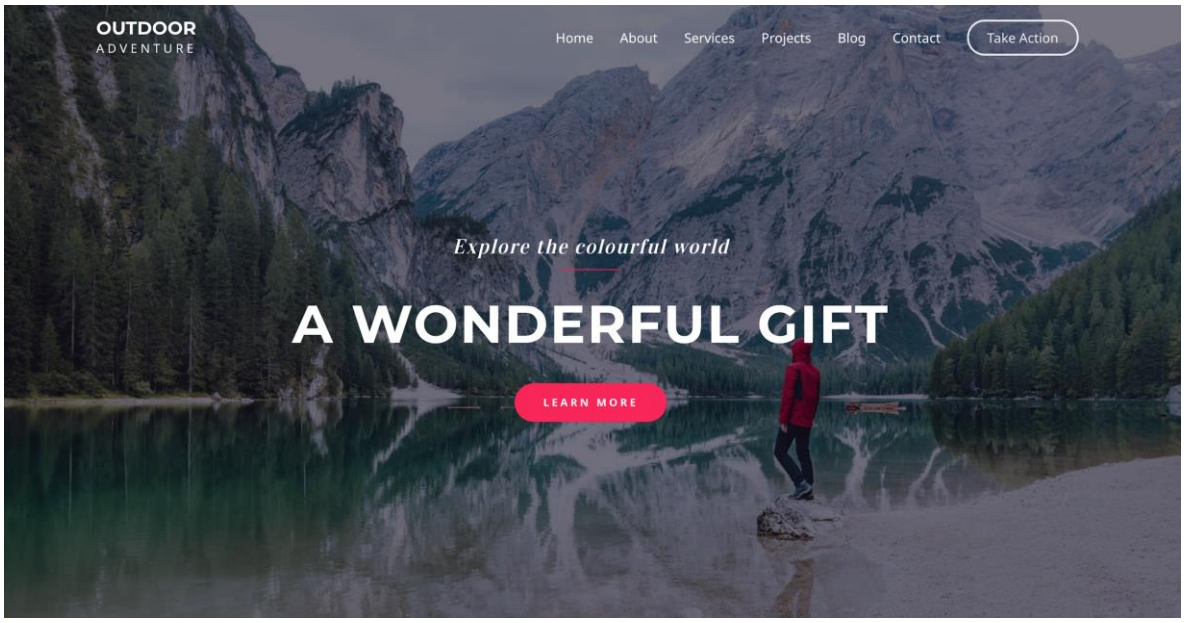
Slika 4.12 Prikaz prozora sa sučeljem web stranice nakon instalacije

Kartica *Izgled*, odnosno teme omogućuju promjenu izgleda web stranice pomoću tema koje su dostupne. Tema predstavlja prezentacijski stil koji u potpunosti može promijeniti izgled web stranice koji je naknadno moguće promijeniti te prilagoditi vlastitim potrebama. Dizajnirane su od strane programera i korisnika, a na raspolaganju ih je na stotine. Dakle, postoji mogućnost promjene teme, a da bi to bilo moguće, potrebno se navigirati na traku Izgled i odabrati stavku Teme gdje je se na raspolaganju nudi veliki broj tema od kojih svaka ima svoje značajke koje čine izgled web stranice. Za potrebe novog web rješenja udruge "Red vitezova Ružice grad" odabrana je Astra tema, nakon aktivacije teme nema velike razlike od početne teme, što je vidljivo na slici 4.13.



Slika 4.13 Prikaz prozora sa sučeljem web stranice nakon instalacije teme Astra

Nakon odabira teme Astra, potrebno je instalirati dodatni *plugin* pod nazivom *Astra starter sites*, koji u svojoj biblioteci sadrži veliki broj tema koje je moguće prilagoditi vlastitim potrebama, bez obzira kakvu je vrstu web aplikacije potrebno razviti. Nakon instalacije, potrebno je izbrisati stranice koje dolaze instalirane sa samim WordPressom, tako da se u administracijskom djelu navigira na karticu Stranice te obriše sve trenutne stranice. Nakon što su izbrisane sve trenutne stranice, moguće je aktivirati jednu od tema koje su dostupne u *Astra starter sites pluginu* tako da se odabere kartica izgled, te nakon toga spomenuti *plugin* gdje je moguće u biblioteci odabrati jednu od dostupnih tema koja najbolje odgovara projektu te zahtjevima klijenta. Na slici 4.14 je vidljiv izgled teme *Outdoor Adventure* koja je pogodna za novo web rješenje udruge.

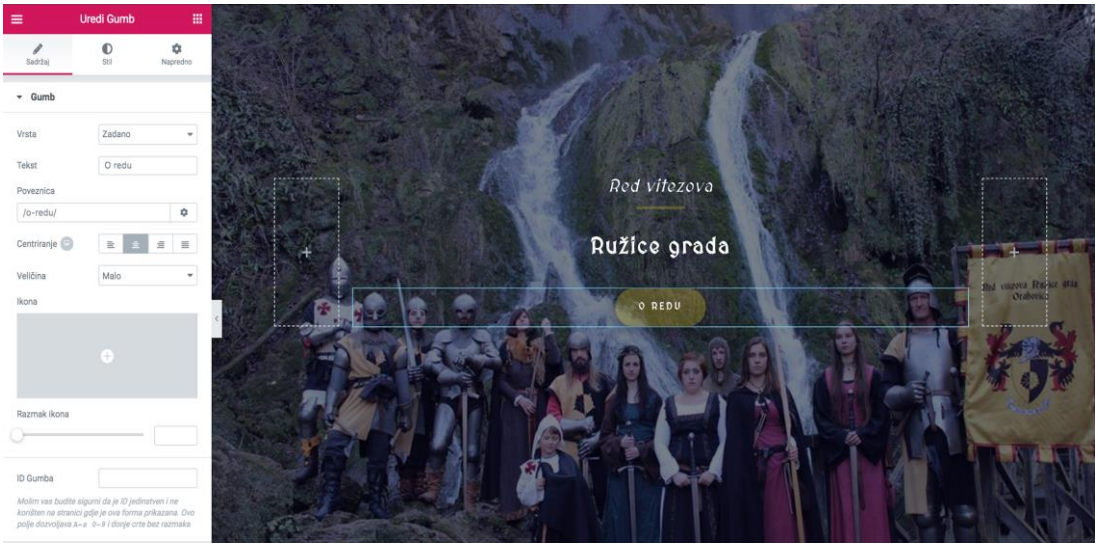


Slika 4.14 Prikaz prozora sa sučeljem web stranice nakon instalacije nove teme

Prednost dodataka koji dolaze uz WordPress je velika, a mnogi od njih olakšavaju izradu sučelja (eng. Frond-end) same web stranice. Jedan od takvih dodataka je alat Elementor koji predstavlja dodatak koji pomaže kod instalacije određene teme, te se naziva page builder. Page builder je dodatak koji omogućuje uređivanje sučelja web stranice, a njegovo korištenje je olakšava izradu tako da djeluje poput blok uređivača (eng. editor) gdje je blokove moguće dovlačiti na sučelje web stranice. Nakon promjene teme WordPress stranice, samo sučelje te cjelokupnu funkcionalnost moguće je prilagoditi vlastitim potrebama. *Page builder* dolazi kao dodatak nakon prve instalacije WordPress-a, no postoji više različitih verzija koje je moguće naknadno instalirati. Za potrebe ovog rada korišten je *Page builder* pod nazivom Elementor. Elementor *Page Builder* mijenja klasični WordPress *editor* tako da je moguće u realnom vremenu mijenjati izgled sučelja te nije

potrebna promjena između samog editora i pregleda (eng. preview). Ovakav alat je izrazito pogodan kod izrade web rješenja zato što je moguće u realnom vremenu s klijentom proći kroz promjene koje bi klijent htio vidjeti na vlastitoj web stranici, što ubrzava sam proces izrade. Instalaciju dodatka je potrebno napraviti u nadzornoj ploči, odnosno administracijskom djelu WordPress stranice gdje se potrebno navigirati na karticu Dodaci, te odabrati opciju Dodaj novi i nakon toga u tražilicu upisati Elementor Page Builder, nakon instalacije je potrebno aktivirati dodatak nakon čega je početna tema kreirana. Općenito, WordPress tema predstavlja skup datoteka koje definiraju izgled web stranice koja je razvijena pomoću WordPress platforme. WordPress tema je kombinacija elemenata, od kojih se sastoji pojedina stranica, ti elementi predstavljaju blokove koji mogu biti bilo što, od različitih widgeta kao što su dugme (eng. *button*), uređivač teksta (eng. text editor), *widget* na koji je moguće postaviti fotografiju ili cijelu fotogaleriju, naslov (eng. heading), različite ikone, te veliki broj drugih funkcionalnosti i elemenata koje je moguće prilagoditi vlastitim potrebama. Naravno, svim stavkama je moguće manipulirati, pristupiti programskom kodu gdje je moguće napraviti promjene na CSS kodu, ako je to potrebno. Kao što je prije rečeno, većina promjena na sučelju web stranice moguće je napraviti pomoću alata Elementor, gdje je moguće u realnom vremenu vidjeti promjene koje su se dogodile na samoj web stranici.

Na slici 4.15 je vidljiv izgled web stranice tijekom korištenja alata Elementor, gdje je u realnom vremenu moguće pristupiti svim elementima na stranici, a s lijeve strane se nalazi izbornik gdje je moguće konfigurirati određeni *widget* ili određeni dio web stranice.Nakon odabira opcije “Uredi s Elementorom” otvara se dio za konfiguraciju sučelja web stranice. Tako je na slici 4.15 prikazan izgled početne stranice nakon otvaranja Elementor-a.



Slika 4.15 Prikaz prozora sa sučeljem web stranice tijekom korištenja alata Elementor

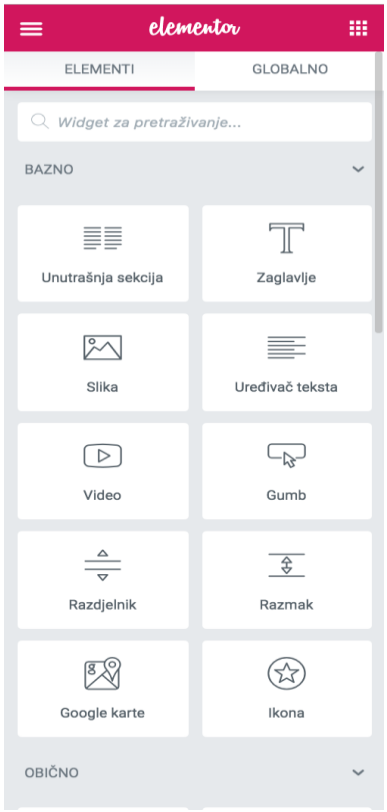
Na slici 4.15 je prikazan pristup gumbu pod nazivom “O redu” koji ima funkciju poveznice koja vodi na stranicu “O redu”. Isto tako služi kao poziv na aktivnost korisnika, što se naziva *Call to action.* Na lijevoj strani ekrana je vidljiv Elementor izbornik koji se sastoji od tri različite kartice. Kartica pod nazivom Sadržaj predstavlja mogućnost upravljanja sadržajem određenog elementa gdje je moguće promijeniti različite funkcionalnosti elementa, a odnose se na sadržaj. Kartica pod nazivom Stil se može usporediti s CSS-om koji služi kao predložak za stil određenog elementa web stranice, tako je pomoću spomenute kartice moguće promijeniti tipografiju, boju elementa kad je statičan i boju kad se na njega navigira sa strelicom miša. Zadnja kartica, pod nazivom Napredno, nudi dodatne mogućnosti, a jedna od stavki koja je dostupna i koja može pomoći kod provođenja testiranja nad tim elementom je CSS ID, gdje je moguće pridodati jedinstveni identifikator elementu te mu kasnije pristupiti kod procesa testiranja.
Nakon odabira opcije, otvara se alat u kojem je moguće uređivati sučelje web stranice, a naziva se Elementor. U njemu je moguće raditi promjene na postojećem izgledu, odnosno na predlošku koji je generiran, no moguće je i dodavati vlastite blokove ili izraditi pojedinu stranicu od početka. Dodavanje novog bloka nalazi se u podnožju stranice, a prikazano je na slikama 4.16 i 4.17.

Slika 4.16 Prozor koji prikazuje način dodavanja novog bloka

Da bi bilo moguće dodati novi *widget,* potrebno je pritisnuti na ikonu znaka za zbrajanje i odabrati blokovski prikaz ili jednostavno dovući *widget* u predviđeni prostor.

Slika 4.17 Prozor koji prikazuje način odabira izgleda blok strukture

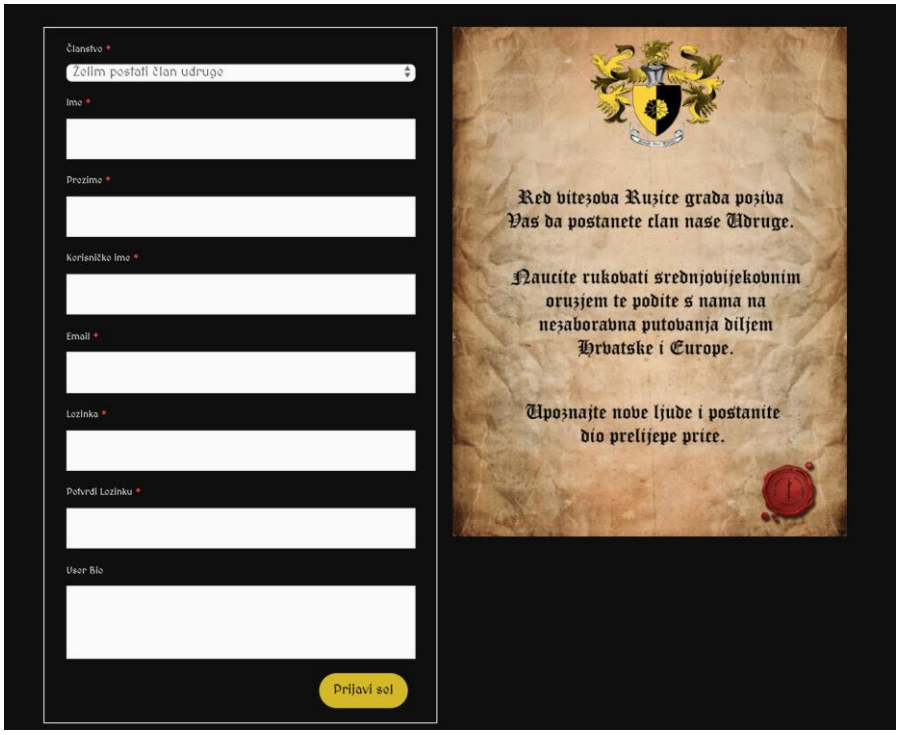
Na slici 4.17 nalazi se opcija za odabir strukture određenog bloka koji definira izgled pojedine stranice ili pojedinog dijela stranice. Nakon kreiranja predloška moguće je početi s uređivanjem same stranice, gdje je moguće izraditi nove stranice ili prilagoditi stranice koje su generirane u predlošku i ima ih 6. Svaka se stranica sastoji od određenih blokova, a svaki se blok sastoji od različitih elemenata koje je moguće prilagoditi vlastitim potrebama. *Widget* je moguće dodati tako da ga se dovuče iz izbornika i pusti na željeno mjesto (eng. drag and drop), izbornik je moguće vidjeti na slici 4.18.



Slika 4.18 Prikaz prozora sa sučeljem alata Elementor

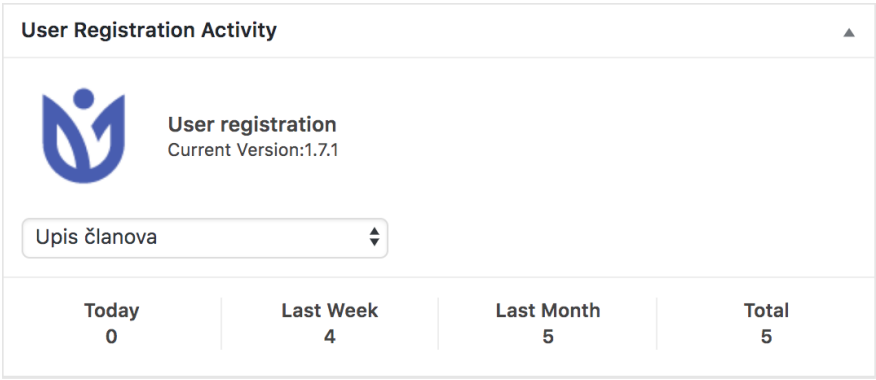
Osim osnovnih elemenata koji se nalaze na slici 4.18, postoji još jedna grupa koja također služi za dodavanje različitih funkcionalnosti na web stranicu. Jedan od takvih elemenata koji je korišten zove se Kratki kod koji predstavlja oznaku određenog elementa, gdje je nakon dodavanja tog elementa potrebno upisati kratki kod u formu s lijeve strane, nakon čega se učitava sadržaj koji je povezan s tom oznakom. Jedna od novih mogućnosti (eng. feature) koja je dio novog web rješenja za udrugu je registracija postojećih ili novih članova udruge, gdje postoji posebna stranica s formom za registraciju pod nazivom Članovi, gdje na početku registracije korisnik mora naznačiti da li je trenutni član udruge pa se želi registrirati u bazu podataka ili je osoba koja bi se željela učlaniti u udrugu, gdje je nakon registracije predstavnik udruge može kontaktirati putem email-a

koji je potrebno upisati u registracijsku formu. Forma za registraciju i prijavu su napravljene pomoću dodatka pod nazivom *User Registration* koji se nalazi u administracijskom djelu WordPress-a gdje je moguće pristupiti podacima korisnika, odnosno administratoru su vidljivi sljedeći podaci: ime, prezime, korisničko ime te email adresa korisnika. Izgled registracijske forme je moguće vidjeti na slici 4.19.



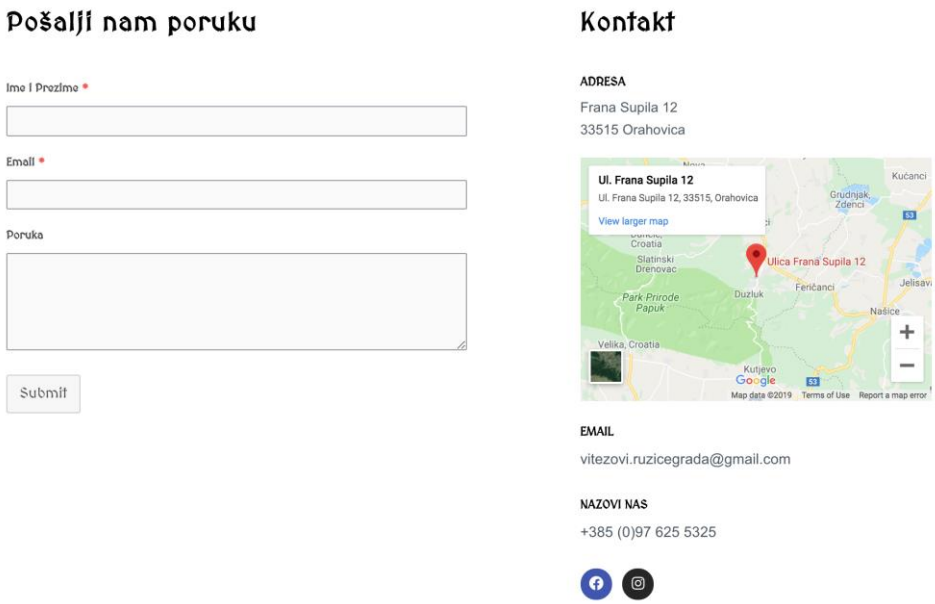
Slika 4.19 Prikaz prozora s formom za registraciju članova udruge

Nakon registracije korisnik dobije email s potvrdom da je registracija uspješna, te se nakon toga može prijaviti da vidi svoj korisnički račun gdje može promijeniti podatke (sve podatke osim korisničkog imena), ako korisnik promijeni podatke, administrator promjene vidi u realnom vremenu. Ovakav način registracije, odnosno učlanjenja u udrugu olakšava cijeli proces, pogotovo ako osoba koja bi željela postati članom ne živi u blizini same udruge gdje mogućnost registracije na samoj web stranici olakšava tu aktivnost. Administrator također može pratiti aktivnosti u pogledu broja registracija te prijava u sustav, gdje je moguće vidjeti ukupan broj registracija, te broj registracija u specifičnom danu, prošlom tjednu ili prošlom mjesecu. Izgled sučelja preko kojeg je moguće pratiti aktivnost, dan je na slici 4.20.



Slika 4.20 Prozor koji prikazuje sučelje za praćenje aktivnosti registriranih korisnika

Ovaj dodatak ima i mogućnost izvlačenja podataka (eng. export) iz administracijskog dijela WordPress stranice gdje administrator mora odabrati opciju za *export* podataka koji su vezani za registraciju korisnika i to u CSV (eng. Comma-Separated Values) formatu. To može biti korisno nakon što se registrira veći broj osoba pa administrator može izvući podatke iz administracijskog djela i koristiti ih tijekom sastanka udruge, gdje može pokazati koliki se broj korisnika registrirao u određenom periodu. Izuzev mogućnosti registracije korisnika, posjetitelj novog web rješenja udruge može kontaktirati udrugu putem email-a, na mobilni uređaj, te preko forme za kontakt, koja se nalazi na stranici Kontakt, na koju je moguće pristupiti pritiskom na gumb Kontakt, koji se nalazi u zaglavlju stranice. Podacima koji su dobiveni korisničkim testiranjem, utvrđeno je da korisnicima smeta što adresa udruge nije prikazana pomoću alata *Google maps*, te je ista funkcionalnost omogućena na novom web rješenju. Također, na stranici Kontakt se nalaze i ikone društvenih mreža Facebook i Instagram, preko kojih je nakon pritiska tipke miša na istu moguće posjetiti stranice udruge na društvenim mrežama. Isto je također dobiveno podacima korisničkog testiranja, gdje je na starom web rješenju udruge bila ikona društvene mreže Facebook, koja nije bila ispravna što je u konačnici korisnicima smetalo, što je vidljivo u podacima korisničkog testiranja. Stranica za kontakt udruge se sastoji od druge boje, odnosno ima drugi stil u usporedbi s ostalim stranicama na novom rješenju web udruge, izgled stranice za kontakt vidljivo je na slici 4.21.

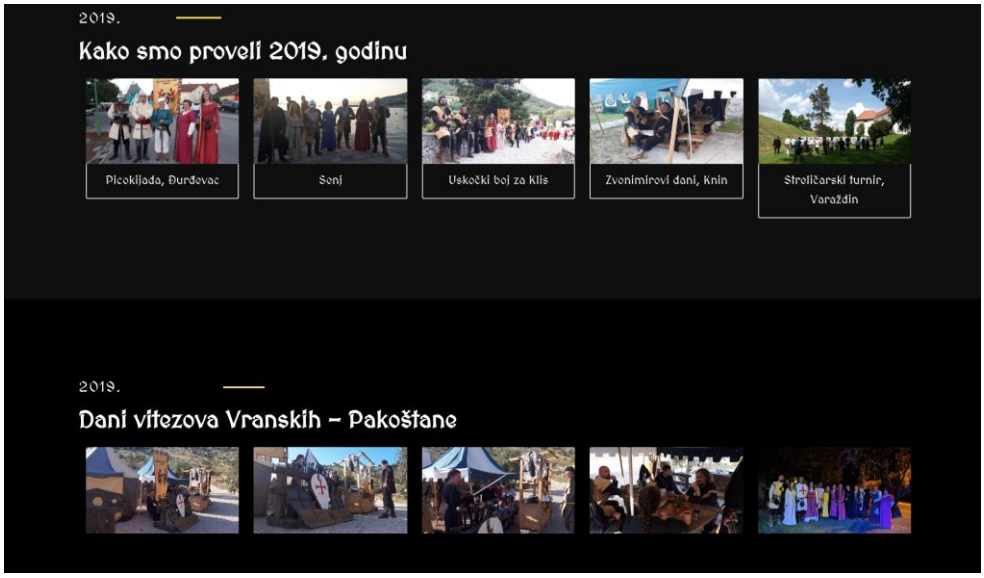


Slika 4.21 Prozor koji prikazuje izgled stranice za kontakt udruge

Kao što je vidljivo na slici 4.21, na stranici za kontakt se nalazi forma za kontakt udruge nad kojom će kasnije biti izvršeno testiranje. Uz formu za kontakt, ova stranica sadrži sve preostale podatke preko kojih je moguće kontaktirati odgovorne u udruzi. Isto tako se na ovoj stranici nalaze dva gumba koji vode na stranice društvenih mreža udruge. Stranica *Galerija* predstavlja stranicu gdje korisnik može vidjeti galerije fotografija s različitih turnira na kojima je udruga bila, gdje je također na temelju podataka korisničkog testiranja promijenjen izgled navigacije prolaska kroz određenu galeriju fotografija, odnosno dodane su navigacijske tipke za prolaz kroz galeriju te tipka za izlazak iz određene galerije, što je vidljivo na slici 4.22.

Slika 4.22 Prozor koji prikazuje mogućnost navigiranja kroz određenu galeriju

Na slici 4.23 dan je izgled stranice fotografija s različitih turnira, također je napravljen pregled 2019. godine s naznačenim turnirom ispod svake od fotografija.



Slika 4.23 Prikaz prozora sa sučeljem stranice *Galerija*

Galerija pod nazivom "Kako smo proveli 2019. godinu" daje dojam da je web rješenje redovito ažurirano, isto tako korisnik može vidjeti gdje je udruga te godine bila, a i moguće je dodati fotografije s turnira koji su se naknadno održali, pomoću alata Elementor. Podaci korisničkog testiranja su također prikazali da je korisnicima u više navrata u starom rješenju smetalo što stranica nije redovito ažurirana, što je vidljivo odmah nakon dolaska na istu, gdje stoji najava sljedećeg turnira, koji se održao prije četiri godine što bi značilo da stranica nije dugo vremena ažurirana.

Stranici na kojoj je moguće vidjeti fotografije udruge s različitih događaja, moguće je pristupiti putem izbornika koji se nalazi u *headeru* web stranice, te s početne stranice gdje se nalaze dva gumba pod nazivom galerija, gdje isti predstavljaju poveznicu koja vodi na stranicu s fotografijama. Taj dio je napravljen na intuitivniji način u usporedbi sa starim web rješenjem, gdje je sama navigacija kroz web stranicu zbunjujuća, što i pokazuju podaci korisničkog testiranja. Jedan od razloga je taj što ne postoji URL struktura i nakon dolaska na specifičnu stranicu, bila to početna stranica ili stranica s kontakt podacima, URL struktura se ne mijenja, a to može predstavljati problem. Iz tog razloga je struktura na novom web rješenju stalna i ne mijenja se s promjenom sadržaja, te je također dodana mogućnost odlaska na stranicu gdje se nalazi galerija fotografija, s početne stranice, što je vidljivo na slici 4.24.

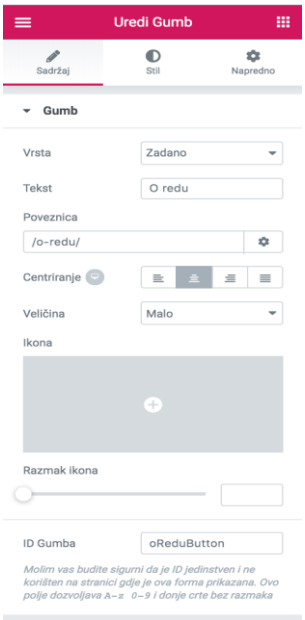
Slika 4.24 Prikaz prozora sa sučeljem početne stanice

Nakon pristupa kodu, vidljivo je da gumb pod nazivom Galerija vodi na stranicu, što je dano u programskom kodu 4.1.

Galerija

Programski kod 4.1 klasa gumba koji vodi na stranicu galerija

Pristupom klasi moguće je automatiziranim testom provjeriti da li spomenuti gumb stvarno vodi na stranicu galerije, što predstavlja plan dohvaćanja elementa. Jedna od prednosti alata Elementor predstavlja mogućnost dodjele ID-a određenom elementu na stranici što olakšava proces lociranja. Nije moguće dodijeliti ID svim elementima na stranici, no elementima koji predstavljaju gumb je. Na slici 4.25 je vidljivo da je gumbu koji vodi na stranicu O redu, dodijeljen ID.



Slika 4.25 Prikaz prozora sa sučeljem alata Elementor

5 TESTIRANJE PROGRAMSKOG RJEŠENJA S ANALIZOM REZULTATA

5.1 Radno okruženje

Radno okruženje predstavlja postavljanje programskog i sklopovskog okruženja za testiranje da bi bilo moguće izvršavati testne skripte koje imaju cilj za potvrditi ispravan rad određenog dijela programske podrške. Drugim riječima, uspješno postavljanje radnog okruženja podrazumijeva konfiguraciju programskog, sklopovskog i mrežnog okruženja u jednu povezanu cjelinu s ciljem uspješnog izvršenja svih testova. Preduvjeti koje je potrebno ispuniti za uspješno izvršavanje testiranja u ovom slučaju predstavljaju instalaciju web-preglednika na kojem je potrebno pokrenuti testne skripte, instalacija svih potrebnih paketa da bi iz terminala bilo moguće pokretati testne skripte, postavljanje datotečne strukture u kojima će se nalaziti testne skripte, te konfiguracija svih datoteka koje će međusobno povezane uspješno pokretati sve testne slučajeve koji će biti napisani s ciljem postizanja uspješnog rada programske podrške.

5.1.1 Node.js + npm

Za početak je potrebno instalirati Node.js pomoću alata NVM (eng. Node Version Manager) koji predstavlja skriptu pomoću koje je moguće upravljati različitim Node.js verzijama. Za instalaciju ili nadogradnju nvm verzije potrebno je u terminal napisati sljedeću naredbu, koja je prikazana prema programskom kodu 5.1.

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash
```

Programski kod 5.1 Prikaz Naredbe *Curl*

Kao što je prikazano, instalacija skripte nvm se izvršava pomoću naredbe *curl (*eng. Client URL*),* koja predstavlja alat za prijenos datoteka koje sadrže URL sintaksu. Podržava veliki broj protokola uključujući HTTP, HTTPS, FTP te mnoge druge [32]. Nakon instalacije je potrebno zatvoriti terminal, ponovno ga pokrenuti te provjeriti da li je nvm uspješno instaliran, s naredbom koja je dana u programskom kodu 5.2.

```
command -v nvm
```

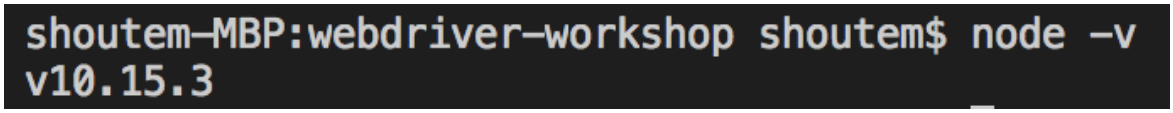
Programski kod 5.2 Prikaz naredbe za provjeru nvm verzije

Nakon uspješne instalacije nvm skripte, potrebno je instalirati posljednju node.js verziju što je moguće upisivanjem sljedeće naredbe u terminal, koja je dana u programskom kodu 5.3.

```
nvm install node
```

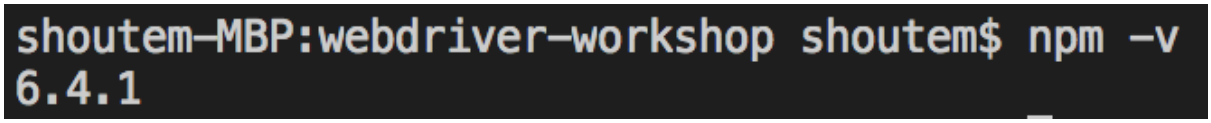
Programski kod 5.3 Prikaz naredbe koja služi za instalaciju posljednje Node verzije

Nakon instalacije je potrebno provjeriti da li je radnja uspješno izvršena, što je prikazano na slici 5.1.



Slika 5.1 Prikaz naredbe za provjeru Node.js verzije u terminalu

Također je potrebno instalirati npm (eng. Node Package Manager) koji sadrži JavaScript biblioteke potrebne za ispravan rad cijelog projekta. Provjera npm verzije je prikazana na slici 5.2



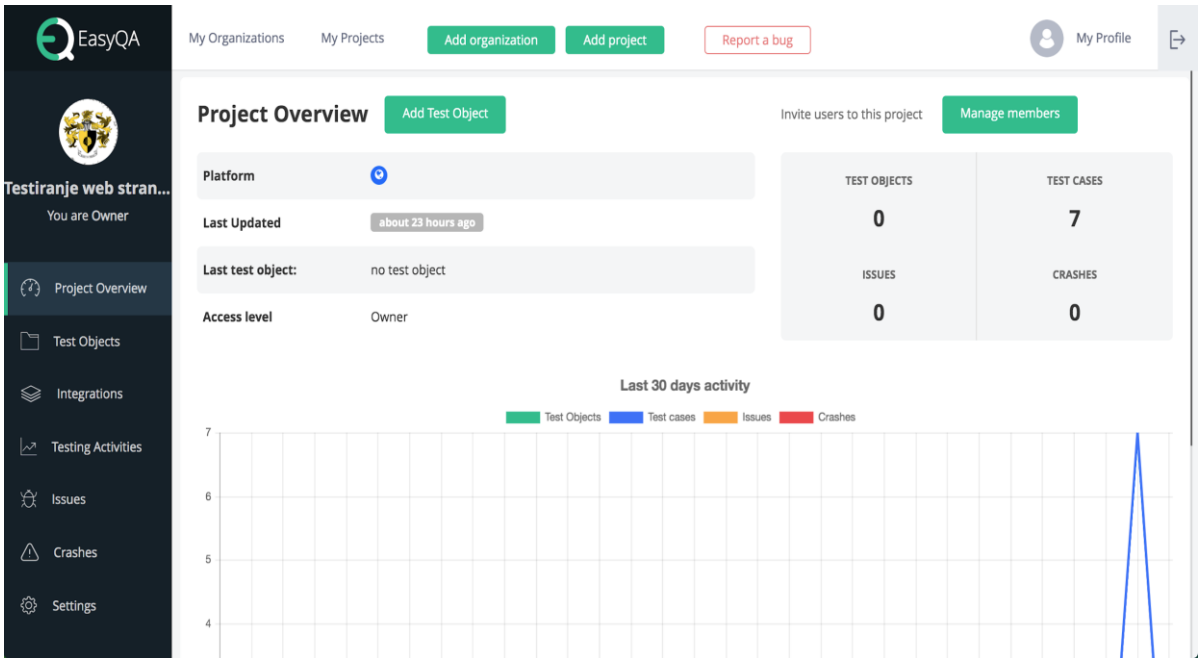
Slika 5.2 Prikaz naredbe za provjeru npm verzije u terminalu

Nakon uspješnog konfiguriranja Node.js okruženja te provjere npm verzije, potrebno je instalirati Visual studio code, u kojem će testne skripte biti napisane, a nakon toga i stvaranja datotečne strukture u kojoj će skripte biti spremljene te u konačnici pokretane iz terminala. Također je potrebno napomenuti da bi se svi paketi trebali nalaziti u istoj datotečnoj strukturi kao i same skripte potrebne za izvršavanje testova.

5.2 Plan testiranja

5.2.1 EasyQA

EasyQA predstavlja alat za upravljanje, odnosno za izradu testnog plana te različitih testnih scenarija s ciljem poboljšanja provedbe procesa testiranja te kvalitete programske podrške. EasyQA je alat koji nije potrebno instalirati, pristup je moguć direktno u web-pregledniku gdje je potrebno napraviti registraciju, prijaviti se na platformu te početi s izradom testnog plana. Nakon prijave je potrebno kreirati novi projekt, te početi s izradom plana samog testiranja. Na slici 5.3 se nalazi izgled sučelja alata EasyQA.

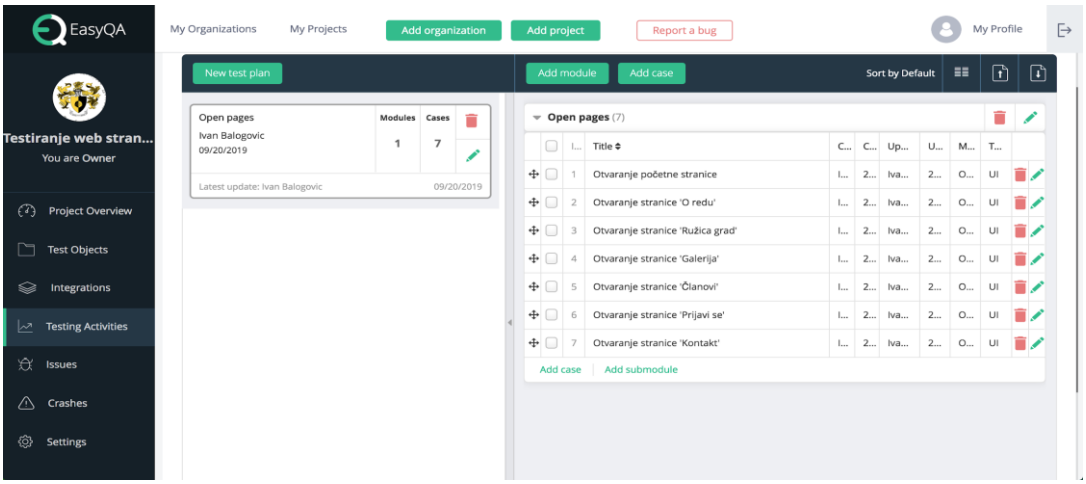


Slika 5.3 Prikaz prozora sa sučeljem alata EasyQA

Testiranje web rješenja udruge će biti podijeljeno na pet različitih modula, koji će biti podijeljeni na nekoliko različitih testnih scenarija:

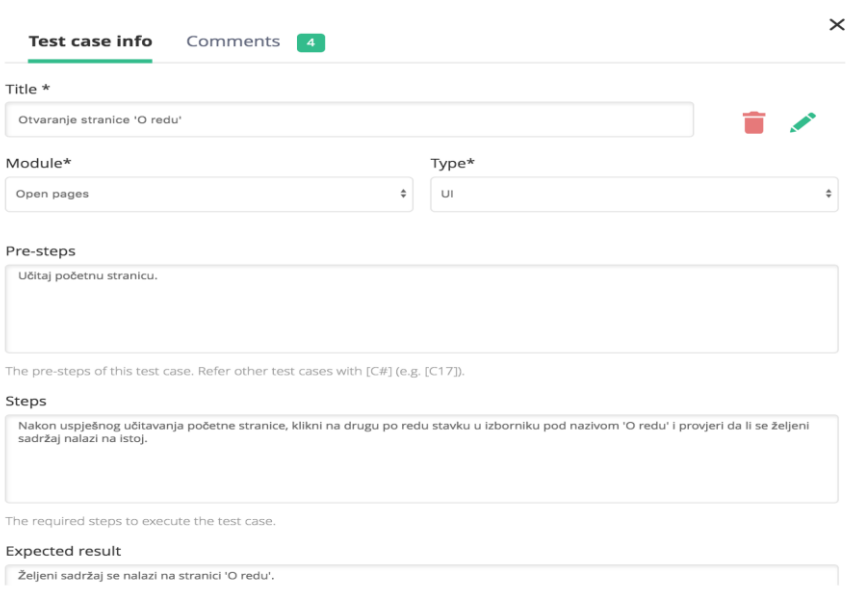
- *Open pages* - cilj ovog testa je da otvori sve pojedine stranice na web rješenju i provjeri da li se određeni sadržaj nalazi na istoj, što predstavlja uspješnost provedenog test.
- *Registration* - cilj ovog testa predstavlja testiranje forme za registraciju novih članova udruge, gdje je potrebno testirati sve pozitivne i negativne ishode kod registracije.
- *Login* - cilj ovog testa predstavlja testiranje forme za prijavu članova koji su izvršili registraciju, gdje je potrebno testirati sve pozitivne i negativne ishode kod prijave.
- *Message form* - cilj ovog testa predstavlja testiranje forme za slanje poruka korisnika koji žele kontaktirati udrugu, gdje je potrebno testirati sve pozitivne i negativne ishode kod slanja poruke.
- *Buttons* - cilj ovog testa predstavlja testiranje svih elemenata koji predstavljaju neku vrstu gumba, koji imaju određenu funkcionalnost.

Na slici 5.4 nalazi se primjer kreiranja testnog plana za web rješenje udruge, koji se sastoji od pet različitih modula, koji predstavljaju testni plan za pojedinačnu funkcionalnost na web stranici.



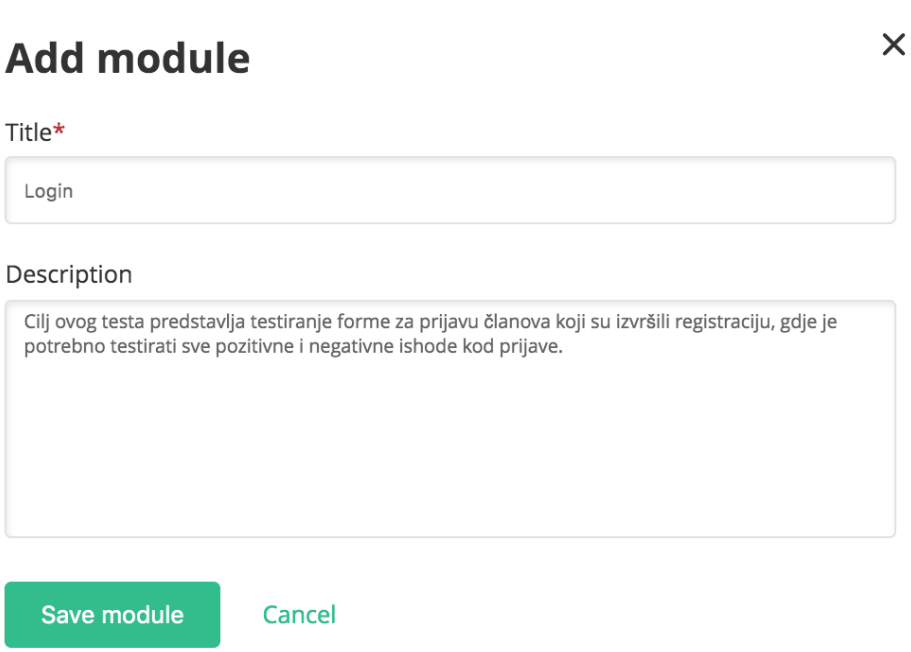
Slika 5.4 Prikaz prozora s testnim planom pod nazivom *Open pages*

Kao što je vidljivo na slici 5.4, testni plan se sastoji od nekoliko modula gdje se svaki modul, odnosno testni plan sastoji od nekoliko testnih scenarija, u ovom slučaju svaki testni scenarij predstavlja određenu radnju kojoj je cilj imitirati korisnika. U svakom testnom slučaju (eng. test case) je potrebno definirati proces provođenja testa, odnosno korake koje bi inače korisnik radio da posjećuje stranicu. Svaki testni scenarij kojeg je moguće definirati putem alata EasyQA sastoji se od više različitih parametara. Svaki testni scenarij mora imati definirano ime, također je potrebno odabrati kojem testnom planu pripada te kojoj vrsti testa pripada, gdje je u ovom slučaju riječ o UI testu, odnosno testu korisničkog sučelja. Isto tako je potrebno definirati korake od kojih se sam test sastoji, gdje koraci predstavljaju radnje koje bi inače korisnik radio, moglo bi se reći da test imitira korisnika gdje se na kraju testa očekuje željeni rezultat. Izgled testa je vidljiv na slici 5.5.



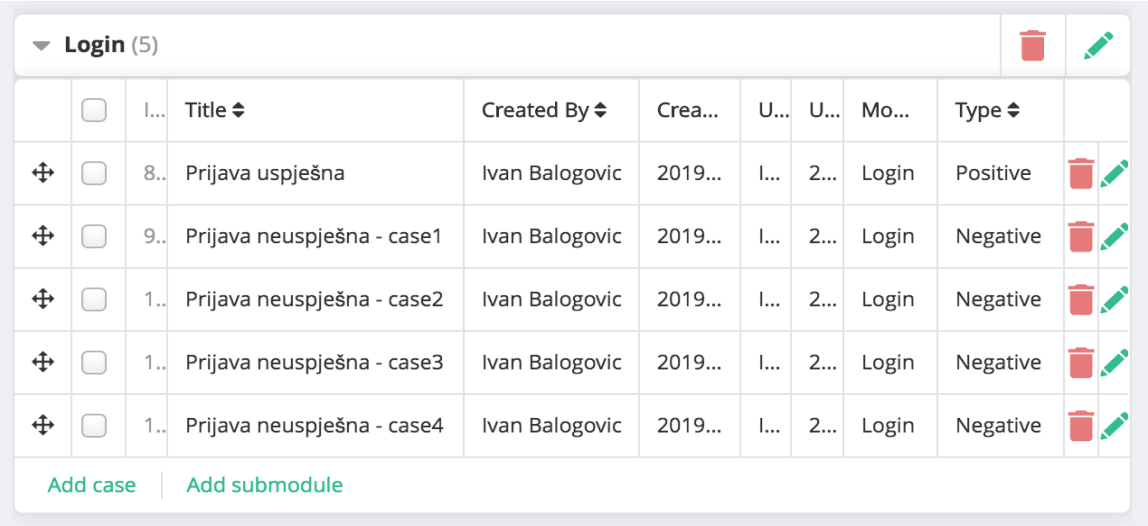
Slika 5.5 Prikaz prozora s primjerom testnog scenarija

Da bi bilo moguće kreirati pojedini modul, koji predstavlja testiranje određenih funkcionalnosti na web stranici, potrebno ga je definirati. Kao što je rečeno, pojedini modul koji je kreiran, predstavlja određenu funkcionalnost. Tako modul *Login* predstavlja test kojem je cilj provjeriti mogućnost prijave korisnika nakon registracije, nakon što se korisnik koji želi postati novi član udruge registrira on s tim podacima ima mogućnost prijave da vidi svoj korisnički račun, gdje je za prijavu potrebno unijeti dvije stavke. Prva stavka u formi za prijavu označava korisničko ime ili email adresu s kojom se korisnik prethodno registrirao, dok druga stavka označava niz znakova koji predstavljaju lozinku koju je korisnik izabrao te upisao kod registracije. Na slici 5.6 prikazan je izgled forme za kreiranje modula koji prezentira aktivnosti koje je potrebno izvršiti kod testiranja funkcionalnosti prijave korisnika u korisnički račun. Uz izradu modula, unutar njega će biti kreirani testni slučajevi koji pokrivaju sve moguće slučajeve koji se mogu dogoditi kod pokušaja korisnika da se prijavi u sustav, koji mogu imati pozitivan i negativan ishod, odnosno uspješan ili neuspješan pokušaj prijave.



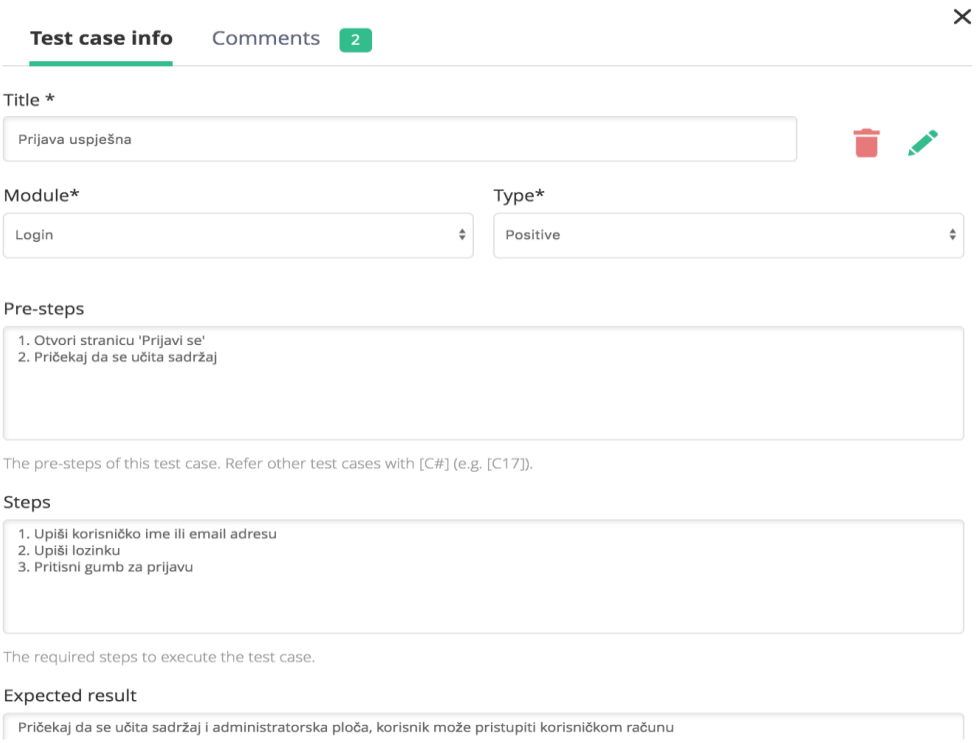
Slika 5.6 Prikaz prozora s primjerom kreiranja modula u alatu EasyQA

Plan testiranja je kreiran u alatu EasyQA, gdje se jedan testni plan, kojeg u alatu EasyQA predstavlja modul sastoji od određenog broja testnih scenarija. Nakon kreiranja plana testiranja za funkcionalnost koja predstavlja prijavu korisnika nakon registracije, izrađeni su svi testni scenariji za istu, koji su prikazani na slici 5.7.



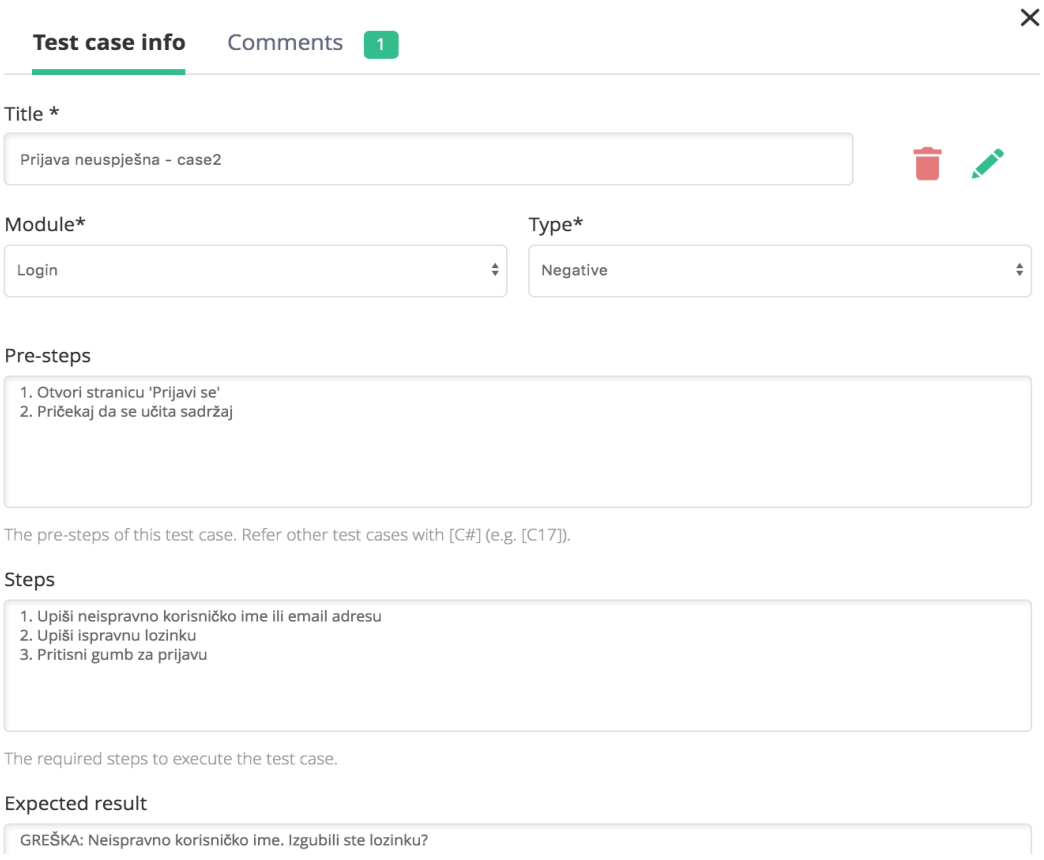
Slika 5.7 Prikaz prozora s planom testiranja za funkcionalnost korisničke prijave u bazu podataka

Kao što je vidljivo na slici 5.7, plan testiranja za funkcionalnost korisničke prijave u bazu podataka se sastoji od pet različitih testnih scenarija gdje prvi testni scenarij predstavlja uspješnu prijavu u sustav od strane korisnika, odnosno pozitivan slučaj. Preostala četiri testna scenarija predstavljaju neuspješnu prijavu u sustav. Na slici 5.8 prikazan je jedan primjer gdje se korisnik uspješno prijavljuje u bazu podataka, a na slici su prikazane sve pojedinosti.



Slika 5.8 Prikaz prozora s testnim scenarijem koji predstavlja uspješnu prijavu u bazu podataka

Osim osnovnih podataka, u testnom scenariju se nalazi stavka *Pre-steps* koja označava koje je korake potrebno poduzeti prije nego li se počne testirati određena funkcionalnost. Tako je prije početka upisa podataka u formu potrebno učitati stranicu na kojoj se nalazi forma, a to je stranica pod nazivom Prijavi se, gdje je potrebno pričekati da se učita određeni sadržaj na stranici. Nakon toga koraka automatska skripta koja se izvodi preko sučelja WebDriver, odnosno web-preglednika Google chrome upisuje podatke u formu, gdje klikom izvršava radnju prijave. Nakon prijave se učitava određeni sadržaj na stranici, koji je definiran te se očekuje u testnoj skripti i ima pozitivan ishod, odnosno označava pozitivan test. Kako je normalno za očekivati da će se korisnik uspješno prijaviti u sustav, isto tako treba očekivati da će biti slučajeva da se korisnik neće moći prijaviti, odnosno upisat će krive podatke u formu za prijavu. U jednom testnom scenariju korisnik može upisati ispravno korisničko ime ili email adresu i neispravnu lozinku, u drugom tesnom scenariju može upisati neispravno korisničko ime ili email adresu i ispravnu lozinku, u trećem slučaju sve neispravne podatke, a u posljednjem testnom scenariju korisnik može ostaviti barem jedno polje prazno te se pokušati prijaviti u sustav. Na slici 5.9 je prikazan jedan od negativnih testnih scenarija.



Slika 5.9 Prikaz prozora s primjerom negativnog testnog scenarija

Kao što je prikazano na slici 5.9, u ovom testnom scenariju korisnik upisuje neispravno korisničko ime ili email adresu, nakon čega je pokušaj prijave u sustav neuspješan. Potrebno je napomenuti da svaki pojedinačni testni scenarij u testnoj skripti predstavlja jednu funkciju, što će biti prikazano kasnije u poglavlju prije čega treba prikazati još nekoliko testnih scenarija koji su napravljeni s ciljem lakšeg pisanja testnih skripti, te u konačnici testiranja kompletnog web rješenja udruge. Najveći broj testnih scenarija sadrži plan testiranja za registraciju novih članova, koji vrše registraciju upisom potrebnih podataka, nakon čega ih se upisuje u bazu podataka, kojoj može pristupiti administrator. Na slici 5.10 su i prikazani, gdje prvi testni scenarij predstavlja uspješnu registraciju korisnika, a preostali negativne ishode kod pokušaja registracije.

User registration(9)

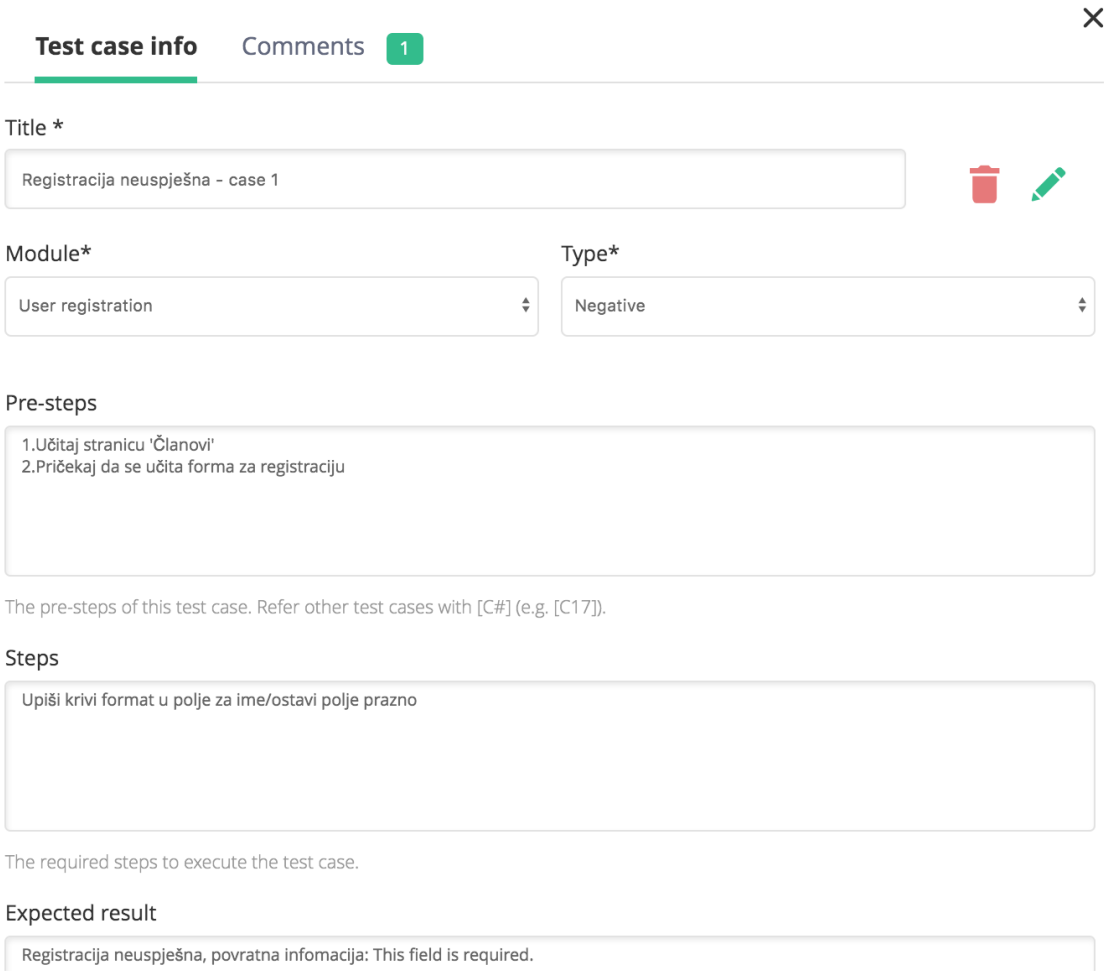
I...	Title	Created By	C...	U...	U...	Module	Type
1...	Registracija uspješna	Ivan Balogovic	2...	I...	2...	User registration	Positive
1...	Registracija neuspješna - case 1	Ivan Balogovic	2...	I...	2...	User registration	Negative
1...	Registracija neuspješna - case 2	Ivan Balogovic	2...	I...	2...	User registration	Negative
1...	Registracija neuspješna - case 3	Ivan Balogovic	2...	I...	2...	User registration	Negative
2...	Registracija neuspješna - case 4	Ivan Balogovic	2...	I...	2...	User registration	Positive
2...	Registracija neuspješna - case 5	Ivan Balogovic	2...	I...	2...	User registration	Negative
2...	Registracija neuspješna - case 6	Ivan Balogovic	2...	I...	2...	User registration	Negative
2...	Registracija neuspješna - case 7	Ivan Balogovic	2...	I...	2...	User registration	Negative
2...	Registracija neuspješna - case 8	Ivan Balogovic	2...	I...	2...	User registration	Negative

Add case | Add submodule

Slika 5.10 Prikaz prozora s planom testiranja za funkcionalnost korisničke registracije u bazu podataka

Kao što je vidljivo na slici 5.10, plan testiranja za funkcionalnost korisničke registracije u bazu podataka se sastoji od devet različitih testnih scenarija gdje prvi testni scenarij predstavlja uspješnu prijavu u sustav od strane korisnika, odnosno pozitivan slučaj. Preostalih osam testnih scenarija predstavljaju neuspješnu prijavu u sustav. Potrebno je napomenuti da su sva polja obavezna, pa je iz tog razloga potrebno napraviti testni scenarij za svaki ishod, odnosno provjeriti što će se dogoditi ako korisnik ne upiše pojedini podatak. Polja za prijavu sadrže ime korisnika, prezime, korisničko

ime, email adresu, lozinku, isto tako korisnik mora još jedanput upisati lozinku kao potvrdu, te na kraju napisati svrhu svoje prijave ili napisati nešto o sebi ne bi li s time olakšao daljnji razgovor s odgovornom osobom u udruzi koja će se povratno javiti korisniku, s ciljem da krene u daljnji proces učlanjenja korisnika u udrugu. Nakon registracije korisnika, administrator udruge dobije email s podacima korisnika koje može iskoristiti za daljnji razgovor s korisnikom, a možda i budućim članom udruge. Na slici 5.11 je prikazan jedan testni scenarij kod registracije korisnika.



Slika 5.11 Prikaz prozora s negativnim testnim scenarijem prilikom registracije korisnika

Testni scenarij koji je prikazan na slici predstavlja slučaj u kojemu korisnik upisuje krivo ime ili ostavlja polje prazno, gdje kod pokušaja registracije dobije određenu povratnu informaciju koja mu daje do znanja da registracija nije uspjela. Preostali plan testiranja za sve funkcionalnosti, a isto tako i svaki pojedini testni scenarij nalaze se u alatu EasyQA, na koji je moguće pristupiti preko bilo kojeg web-preglednika, pa nema potrebe prikazivanja svih slučajeva. Neki od testnih scenarija koji nisu prikazani u ovom poglavlju, bit će prikazani kod prikaza samog testiranja, kasnije u radu.

5.3 WebDriver API

Selenium 2.0 ima prednost koja ga čini moćnim alatom za testiranje, a to je integracija WebDriver API-ja. WebDriver je dizajniran da pruži jednostavnije programsko sučelje kojem je cilj rješavanje nekih od ograničenja koje je imao Selenium-RC API, gdje je WebDriver razvijen za bolje rukovanje web stranicama gdje se elementi mogu promijeniti bez da se sama stranica ponovno učita. Nadalje, cilj alata WebDriver je pružanje dobro dizajniranog te objektno-orijentiranog API-ja koji pruža bolju podršku za sve probleme s kojima se može susresti kod testiranja naprednih web aplikacija. Selenium WebDriver šalje direktne zahtjeve web-pregledniku koristeći sve njegove prednosti te podršku za automatizaciju, a kako se ti zahtjevi šalju, te koje značajke oni podržavaju ovisi o web-pregledniku s kojim se komunicira [33].

5.3.1 Implementacija

Za početak procesa testiranja potrebno je pripremiti datotečnu strukturu projekta, odnosno putanju gdje će sve skripte i konfiguracijske datoteke biti sačuvane. Potrebno je kreirati novu mapu (eng. Folder) pomoću naredbe koja je prikazana u programskom kodu 5.4, koju je potrebno izvršiti u terminalu.

```
mkdir WebDriver-workshop
```

Programski kod 5.4 Prikaz naredbe koja služi za kreiranje nove mape

Pomoću naredbe *mkdir* moguće je kreirati novu mapu iz terminala, nakon čega se pomoću naredbe cd potrebno pozicionirati u direktorij u koji se želi prijeći. Nakon kreiranja potrebne datotečne strukture, potrebno je postaviti Node.js projekt, gdje naredba koja je prikazana prema programskom kodu 5.5 služi za inicijalizaciju praznog NMP projekta.

```
npm init -y
```

Programski kod 5.5 Prikaz naredbe koja služu za inicijalizaciju Node.js projekta

Da bi WebDriverIO bio integriran u testni paket, potrebno ga je instalirati lokalno prema naredbi koja je dana u programskom kodu 5.6.

```
npm i WebDriverio --save-dev
```

Programski kod 5.6 Prikaz naredbe koja služi za lokalnu instalaciju WebDriverIO projekta

Nakon toga je potrebno instalirati WebDriver cli (eng. Command Line Interface) pomoću naredbe koja je dana u programskom kodu 5.7.

```
npm i @wdio/cli --save-dev
```

Programski kod 5.7 Naredba koja služi za instalaciju WebDriver cli-a

Sljedeći korak je generiranje konfiguracijske datoteke, u kojoj će biti spremljene sve WebDriverIO postavke. Taj korak je dan u obliku programskog koda 5.8.

```
./node_modules/.bin/wdio config
```

Programski kod 5.8 Naredba koja služi za generiranje konfiguracijske datoteke

Skripte pisane za testiranje će biti izvršene lokalno, a *framework* korišten za njihovo izvršavanje u web-pregledniku bit će mocha. Mocha predstavlja *framework* za pokretanje JavaScript datoteka u web-pregledniku po izboru. Instalacija je moguće izvršiti pomoću naredbe koja je dana u programskom kodu 5.9.

```
npm install --save-dev mocha-WebDriver
```

Programski kod 5.9 Naredba koja služi za instalaciju moche

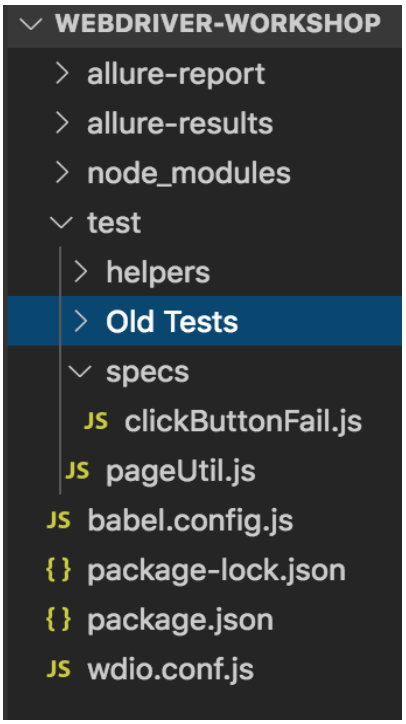
5.3.2 Selenium WebDriver u programskom jeziku JavaScript

Testiranje novog web rješenja udruge "Red vitezova Ružice grada" izvršeno je u Visual studio code-u, upotrebom programskog jezika JavaScript. Svaku kreiranu skriptu potrebno je pohraniti s dodatkom .js, kako bi se mogla izvršavati iz terminala, koji je ugrađen u Visual studio code. Ako se omogući korištenje terminala unutar alata Visual studio code, potrebno se u terminalu navigirati u specifični direktorij pomoću naredbe cd, odnosno u onaj direktorij u kojemu se nalazi skripta. Da bi bilo moguće pokretati skripte iz datotečnog direktorija koji je prethodno određen potrebno je dodati putanju u datoteku koja služi za postavke WebDriver-a, koja je prethodno kreirana, putanja je dana u programskom kodu 5.10.

```
specs: [ './test/specs/**/*.js' ],
```

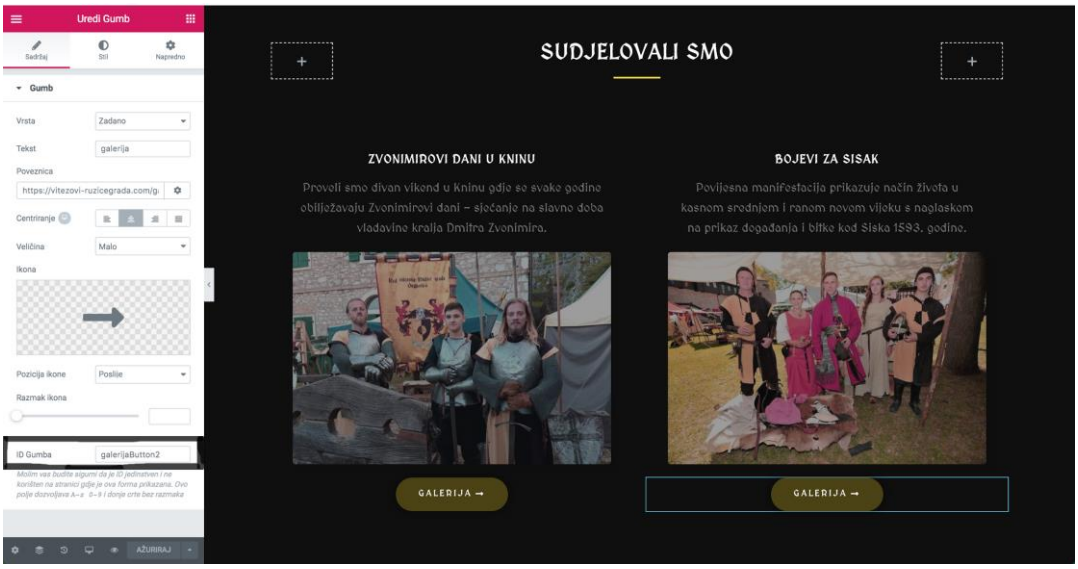
Programski kod 5.10 Dodavanje putanje u konfiguracijsku datoteku

Izgled datotečne strukture u alatu Visual studio code prikazan je na slici 5.13.



Slika 5.13 Prikaz datotečne strukture projekta u alatu Visual studio code

Kao što je prikazano na slici 5.13, a definirano u postavkama WebDriver-a svi testovi se nalaze u mapi pod nazivom test, a onaj test koji je potrebno izvršiti treba se nalaziti u mapi pod nazivom specs. Test pod nazivom *clickButtonFail.js* prikladno je prikazati kao prvi primjer zato što je izvršen tijekom razvoja web rješenja. Na početnoj stranici novog web rješenja implementirana su dva gumba koji se nalaze ispod fotografija koje predstavljaju određeni turnir na kojima je udruga bila sudionik. Njihova funkcija je da nakon što korisnik klikne na gumb vode na stranicu na kojoj se nalaze galerije fotografija različitih turnira. Agilni razvoj nalaže da se web rješenje testira tijekom samog razvoja, s ciljem uštede vremena te osiguranja kvalitete programske podrške te je to i učinjeno. Cilj testa je testirati dva gumba koji vode na stranicu galerije. Da bi bilo moguće pristupiti tim elementima, potrebno je pronaći i dohvatiti određeni element prema zadanom parametru, gdje WebDriver pruža veliki broj metoda koje omogućuju pristup tim elementima. Također jedan od razloga za odabir alata Elementor *page builder* koji služi za pogodniji razvoj WordPress stranice, bila je i mogućnost pristupa svim elementima na web stranici, odnosno na njenom sučelju. Elementor pruža mogućnost dodjeljivanja *Id* atributa određenom elementu, a preko kojeg je lakše pristupiti tom elementu. Dodjeljivanje *Id* atributa elementu koji se nalazi na web stranici u obliku gumba izvršava se preko alata Elementor, a prikazano je na slici 5.14.



Slika 5.14 Prikaz prozora sa postupkom dodjeljivanja *Id* atributa gumbu galerija

Nakon dodjele *Id-a* elementu koji predstavlja gumb, pod nazivom *galerijaButton2*, u programskom kodu je moguće definirati varijablu kojoj se dodjeli vrijednost *Id* elementa, a ta se varijabla kasnije koristi za pristup te sve aktivnosti koje oponašaju radnje korisnika. Isto tako je moguće pristupiti klasi ili identifikatoru elemenata preko radnje *inspect*, u samom web-pregledniku. Način na koji je definiran identifikator, prikazan je u programskom kodu 5.11.

```
const galerijaButton2Id = "#galerijaButton2";
```

Programski kod 5.11 Dodjeljivanje identifikatora elementu gumb

Na isti način je definiran identifikator drugog gumba, koji se nalazi na istoj stranici. Proces dohvaćanja elementa počinje s učitavanjem početne stranice, što je definirano prema kodu 5.12.

```
describe("Open links from home page", function() {
beforeEach(function() {
browser.url("https://vitezovi-ruzicegrada.com");
$("#content").waitForExist();
});
```

Programski kod 5.12 Prikaz programskog koda koji služi za učitavanje početne stranice

Kao što je prikazano u programskom kodu 5.12, funkcija *browser.url* dohvaća URL početne stranice udruge te pomoću funkcije *waitForExist* čeka da se učita sadržaj na istoj, odnosno da se učita bilo koji element koji se nalazi unutar DOM (eng. Document Object Model) strukture. Nakon

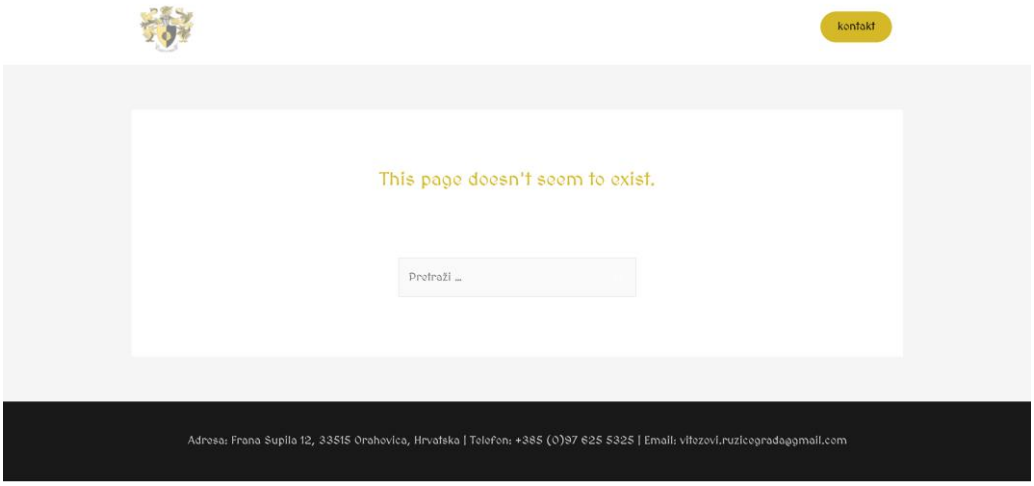
što dohvati element skripta kreće s izvršavanjem prvog testnog scenarija. Potrebno je napomenuti da skripta prije svakog sljedećeg testnog scenarija koji se nalazi u skripti, učitava početnu stranicu, što omogućuje funkcija *beforeEach.*

Nakon učitavanja početne stranice potrebno je locirati spomenuti gumb koji vodi na stranicu galerije i izvršiti radnje koje oponašaju korisnika, što je prikazano prema programskom kodu 5.13.

```
it("should open page Gallery page with button1", function() {
$(galerijaButton1Id).waitForDisplayed();
$(galerijaButton1Id).click();
$(galerijaTitleFail).waitForDisplayed();
compareUrls(galerijaUrl);
});
```

Programski kod 5.13 Testni scenarij koji provjerava otvaranje stranice galerija

Programski kod 5.13 prikazuje testni scenarij koji provjerava da li se stranica galerije otvori nakon što korisnik klikne na gumb koji bi trebao voditi na istu. Funkcija *waitForDisplayed* očekuje da se element čiji je identifikator predan, a predstavlja *Id* gumba - pojavi na početnoj stranici. Drugoj po redu funkciji *waitForDisplayed* je predana klasa koja označava naslov koji se pojavi na stranici koja se otvori nakon klika na gumb, taj naslov je vidljiv na slici 5.15. S obzirom na to da je provjerom prije samog testa utvrđeno da elementi koji predstavljaju gumb ne vode na stranicu galerije, ovaj test je napisan s ciljem potvrđivanja istog, što je potvrdila funkcija koja uspoređuje očekivani URL i onaj koji je učitan. Na slici 5.15 prikazana je poruka koja se pojavi nakon pritiska na gumb koji bi trebao voditi na stranicu galerije, gdje je prikazano da tražena stranica ne postoji.



Slika 5.15 Prikaz prozora sa porukom da tražena stranica ne postoji

Kao što je vidljivo na slici, gumb koji bi trebao voditi na stranicu galerije s fotografijama turnira na kojima je udruga sudjelovala ne odrađuje svoju funkciju. Nakon što je testom potvrđeno da gumb ne vodi na stranicu galerije, isto je potrebno promijeniti alatom Elementor, tako da se promjeni URL poveznica stranice na koju bi gumb trebao voditi. Poslije promjene je potrebno izvršiti novi test, s ciljem potvrđivanja funkcionalnosti koju bi gumb Galerija trebao imati. Uz testiranje gumba koji vodi na stranicu galerije, na početnoj stranici se nalazi gumb koji vodi na stranicu O redu, te gumb koji vodi na stranicu Kontakt koje je također potrebno testirati. Skripta za testiranje se sastoji od testnih scenarija koji očekuju pozitivan odziv, odnosno otvaranje stranice na koju bi gumb trebao voditi. Kod 5.14 prikazuje testni scenarij koji provjerava da li gumb "O redu" vodi na istoimenu stranicu.

```
it("should open O Redu page with a button", function() {
$(oReduButtonId).waitForDisplayed();
$(oReduButtonId).click();
$(oReduTitle).waitForDisplayed();
compareUrls(oReduUrl);
});
```

Programski kod 5.15 Prikaz programskog koda koji služi za učitavanje stranice "O redu" pomoću gumba

Prva linija u testnom scenariju predstavlja funkciju koja čeka da element bude prikazan, a pristupa mu preko identifikatora koji je definiran u početku skripte. Druga linija predstavlja funkciju koja klika na gumb, dok treća linija čeka da se učita stranica "O redu", što provjerava pristupom elementu koji je vezan za naslov na stranici, pristup elementu je omogućen definiranjem klase tog elementa. Posljednja funkcija uspoređuje URL trenutne stranice, s URL-om koji je definiran na početku skripte.

Na novom web rješenju udruge omogućena je registracija korisnika, s ciljem lakšeg učlanjenja u udrugu, gdje posjetitelji mogu ispuniti formu za registraciju. Nakon uspješne registracije, potencijalni član se upisuje u bazu podataka, nakon čega ga odgovorna osoba u udruzi može kontaktirati te nastaviti s procesom učlanjenja u udrugu. Sam proces registracije testiran je s devet različitih testnih scenarija s ciljem utvrđivanja ispravnosti registracijske forme. Da bi formu bilo moguće testirati, potrebno je locirati elemente koji su povezani za svako pojedino polje u registracijskoj formi. Isto tako je potrebno definirati sve vrijednosti koje će biti upisane u polja koja je potrebno ispuniti da bi registracija bila uspješno izvršena. Prvi testni scenarij predstavlja slučaj u kojem je registracija uspješno izvršena, gdje su svi uneseni podaci valjani. Preostalih osam

testnih scenarija predstavlja slučajeve u kojima je barem jedna vrijednost nevaljana ili je polje za unos prazno. Svaki pojedini testni scenarij je definiran u istoj skripti, a proces kojim se skripta izvršava je takav da se nakon svakog pokušaja registracije stranica ponovno učita nakon čega se izvršava sljedeći testni scenarij, funkcija koja to omogućuje dana je prema programskom kodu 5.16.

```
describe("Open Pages for user register", function() {
beforeEach(function() {
browser.url("https://vitezovi-ruzicegrada.com/clanovi/");
$(signUpFormId).waitForExist();
});
```

Programski kod 5.16 Prikaz programskog koda sa funkcijom koja omogućuje ponovno učitavanje stranice

Nakon što se učita stranica za registraciju, preko testne skripte se izvršava svaki testni scenarij gdje se između svakog novog pokušaja registracije ponovno učita stranica funkcijom koja je prikazana u kodu 5.16. Izgled testnog scenarija bez email adrese, prikazan je u kodu 5.17.

```
it("should sign up", function() {
$(signUpFormId).scrollIntoView()
$(signUpFormId).waitForExist();
$(signUpFormId).scrollIntoView();
$(imeid).setValue(ime);
$(prezimeid).setValue(prezime);
$(korImeid).setValue(korIme);
$(emailid).setValue();
$(lozinkaid).setValue(lozinka);
$(potvrdiLozinkuid).setValue(potvrdiLozinku);
$(userBioid).setValue(userBio);
utilPage.clickButton(loginButton);
 browser.pause(5000);
utilPage.wait(emailFailId);
 });
```

Programski kod 5.17 Prikaz programskog koda koji služi za upisivanje podataka u polja registracijske forme

Potrebno je napomenuti da su funkcije pod nazivom *utilPage.clickButton(); i utilPage.wait();* pozvane iz druge skripte, koja se nalazi u istoj datotečnoj strukturi kao i skripta za testiranje registracijske forme. Svaki preostali testni scenarij se razlikuje jedino po nedostatku argumenta za određeno polje, dok u slučaju skripte koja je dana u programskom kodu 5.17 argument koji nedostaje email adresa.

Ostatak plana testiranja koji je proveden s ciljem osiguravanja kvalitete programske podrške sastoji se od skripte koja omogućuje provjeru otvaranja svih stranica od kojih se web rješenje sastoji koje su, osim početne, stranica pod nazivom "O redu" u kojoj se nalaze informacije o udruzi, stranice pod nazivom "Ružica grad" u kojoj je moguće pronaći više informacija o istoimenoj srednjovjekovnoj utvrdi koja se nalazi u blizini gradića Orahovica. Sljedeća stranica koja je dio novog web rješenja je stranica pod nazivom "Galerija" u kojoj se nalaze galerije fotografija s turnira u kojima je udruga bila sudionik. Stranica pod nazivom "Članovi", te "Prijavi se" predstavljaju stranice za registraciju potencijalnih članova udruge, odnosno za prijavu u korisnički račun. Posljednja stranica koja se nalazi na novom web rješenju predstavlja stranicu za kontakt, gdje se nalazi forma za kontaktiranje udruge, te osnovne informacije o udruzi te poveznice za društvene mreže udruge. Osim testiranja valjanosti registracijske forme, te funkcionalnosti koje predstavljaju gumbi na početnoj stranici, proces testiranja je izvršen nad kontakt formom kojoj je namjena slanje poruke udruzi od strane korisnika, formi za prijavu u korisnički račun te gore navedena funkcionalnost otvaranja svih pojedinačnih stranica koje se nalaze na novom web rješenju udruge. Sve spomenute skripte se nalaze u obliku programskog koda u prilogu.

5.4 Rezultati testova sa analizom

5.4.1 Allure

Rezultati testiranja su prikazani u obliku izvješća pomoću alata Allure gdje je svako izvješće izvršeno u dva koraka. Tijekom izvršavanja skripte, što označava prvi korak, biblioteka pod nazivom *adapter* koja je ugrađena u *framework* za testiranje sprema informacije o izvršenom testu u XML (eng. EXtensible Markup Language) datoteku. Tijekom generiranja izvješća, XML datoteka se prevodi u HTML izvješće, koje se izvršava u web-pregledniku gdje je moguće vidjeti izvještaj svih provedenih testova. Allure je moguće povezati s WebDriverom bez obzira na programski jezik u kojemu su testne skripte pisane. Da bi Allure bilo moguće koristiti potrebno ga je povezati s preostalim datotekama koje su neophodne za pravilno izvršavanje testnih skripti. Prvo

je potrebno dodati programski kod u datoteku package.json, koji je prikazan prema programskom kodu 5.18.

```
{
"devDependencies": {
"@wdio/allure-reporter": "^5.0.0"
}
}
```

Programski kod 5.18 Programski kod koji služi za uključivanje Allure-a u package.json

Instalacija alata se vrši preko terminala, pomoću naredbe koja je prikazana u programskom kodu 5.19.

```
npm install @wdio/allure-reporter --save-dev
```

Programski kod 5.19 Naredba koja služi za instalaciju alata Allure

Zadnji korak predstavlja konfiguracija direktorija u kojem će se nalaziti XML datoteke, koje predstavljaju izvješća testiranja, gdje je potrebno dodati sljedeći programski kod u skriptu wdio.conf.js.

```
exports.config = {
// ...
reporters: [['allure', {
outputDir: 'allure-results',
disableWebDriverStepsReporting: true,
disableWebDriverScreenshotsReporting: true,
}]],
 // ...
}
```

Programski kod 5.20 Programski kod koji služi za konfiguraciju direktorija

Da bi bilo moguće generirati izvješća, potrebno je dodati sljedeću liniju na početak svake testne skripte, koja je dana u programskom kodu 5.21.

```
import allureReporter from '@wdio/allure-reporter'
```

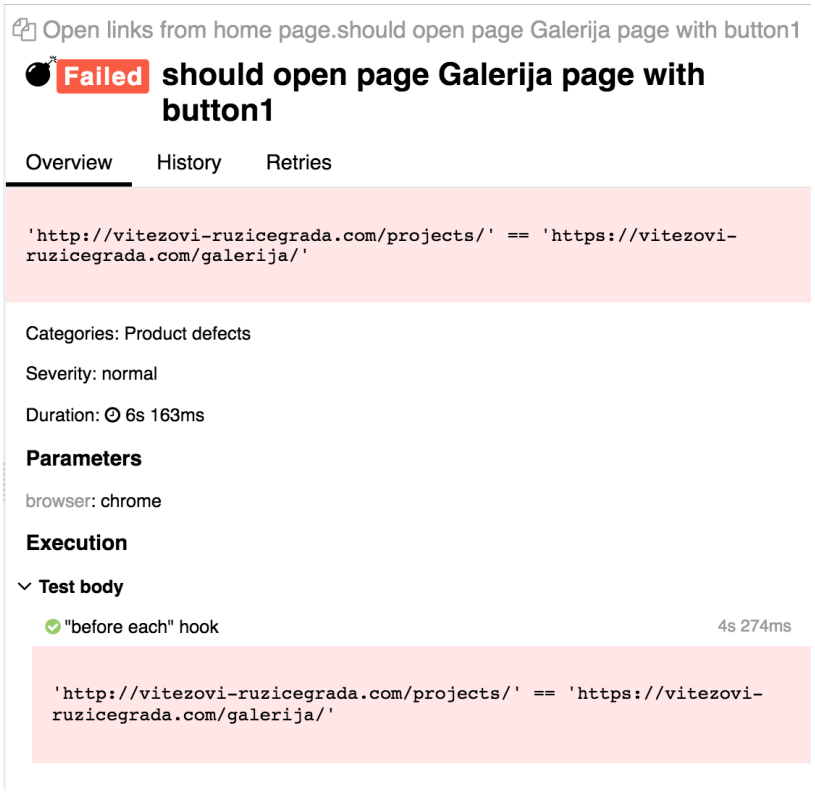
Programski kod 5.21 Programski kod koji služi Dodavanje Allure-a u skriptu

Nadalje, potrebno je dodati i sljedeću liniju u svaki testni scenarij za koji treba izvještaj. Naredba je dana u programskom kodu 5.22.

```
allureReporter.addFeature();
```

Programski kod 5.22 Dodavanje Allure-a testu

Nakon konfiguracije, Allure generira izvješće za svaku skriptu, odnosno za svaki testni scenarij u koji je uključen. Sljedeće izvješće koje je Allure generirao odnosi se na testni slučaj gdje testna skripta provjerava da li se stranica galerije otvori nakon što korisnik klikne na gumb galerija, koji se nalazi na početnoj stranici. S obzirom na to da je provjerom prije samog testa utvrđeno da elementi koji predstavljaju gumb ne vode na stranicu galerije, ovaj test je napisan s ciljem potvrđivanja istog. Što je izvješće i pokazalo. Izvješće o ishodu izvršenja skripte za prvi gumb prikazano je na slici 5.16.

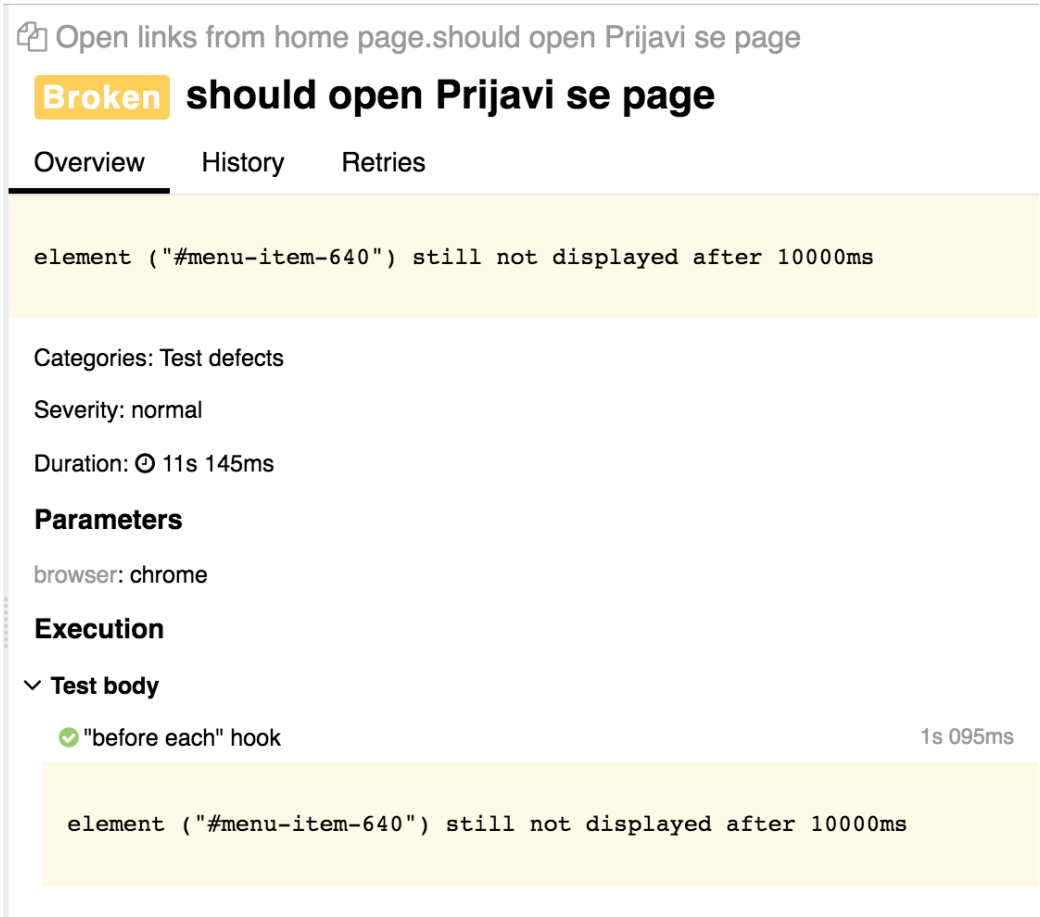


Slika 5.16 Prikaz prozora koji se sastoji od izvještaja generiranog alatom Allure

Na slici je vidljivo da je glavna stavka testa područje gdje skripta uspoređuje URL poveznice dvije različite stranice s web rješenja udruge. S lijeve strane se nalazi URL pogrešne stranice, a sa desne strane se nalazi URL stranice galerija, što je bio očekivani ishod nakon klika na gumb galerija. Na sljedećoj slici se nalazi izvješće o ishodu izvršenja skripte za drugi gumb što je i prikazano je na slici 5.17.

Slika 5.16 Prikaz prozora koji se sastoji od izvještaja generiranog alatom Allure

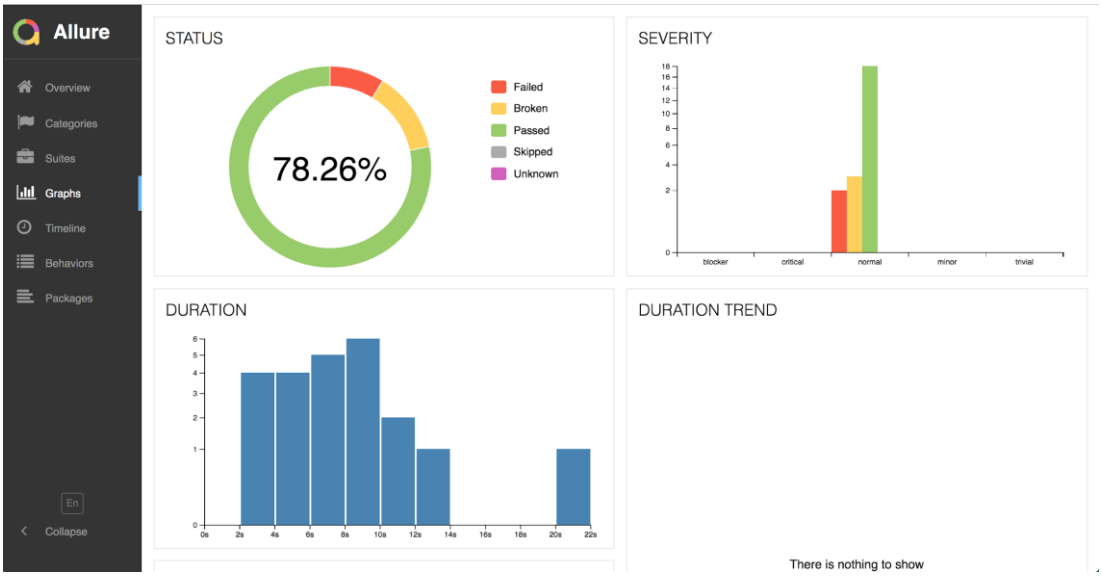
Allure predstavlja alat koji prikazuje grafički i analitički prikaz podataka koji su generirani izvršavanjem testnih skripti, s ciljem dobivanja što boljeg uvida u uspješnost procesa testiranja određene programske podrške. XML datoteke koje Allure generira moguće je obrisati, što je izrazito korisno ako je potrebno analizirati podatke određenog testnog scenarija. Još jedna od prednosti alata je ta što je moguće u realnom vremenu vidjeti izvješće pojedinog testa, na vrlo jednostavan način. Uz to što je moguće pristupiti izvješću svakog pojedinog testnog scenarija, također je moguće vidjeti i podatke cijelog testnog plana uz brojčani prikaz koji se dijeli na testne scenarije koji su uspješno izvršeni, odnosno imaju pozitivan ishod. Isto tako je prikazan broj testnih scenarija koji imaju negativan ishod, odnosno funkcionalnost određenog elementa nije očekivana. Postoji i mogućnost da se parametar koji je određen kao identifikator određenog elementa ne pojavljuje na stranici. Takav slučaj je prikazan na slici 5.17, a odnosi se na testni scenarij koji provjerava funkcionalnost forme za prijavu korisnika na korisnički račun, gdje skripta neuspješno pokušava locirati element "Prijavi se", a potreban je da bi bi se došlo na samu stranicu za prijavu korisnika. Element se nalazi u glavnom izborniku. Navedeno izvješće se nalazi na slici 5.17.



Slika 5.17 Prikaz prozora koji se sastoji od izvještaja generiranog alatom Allure

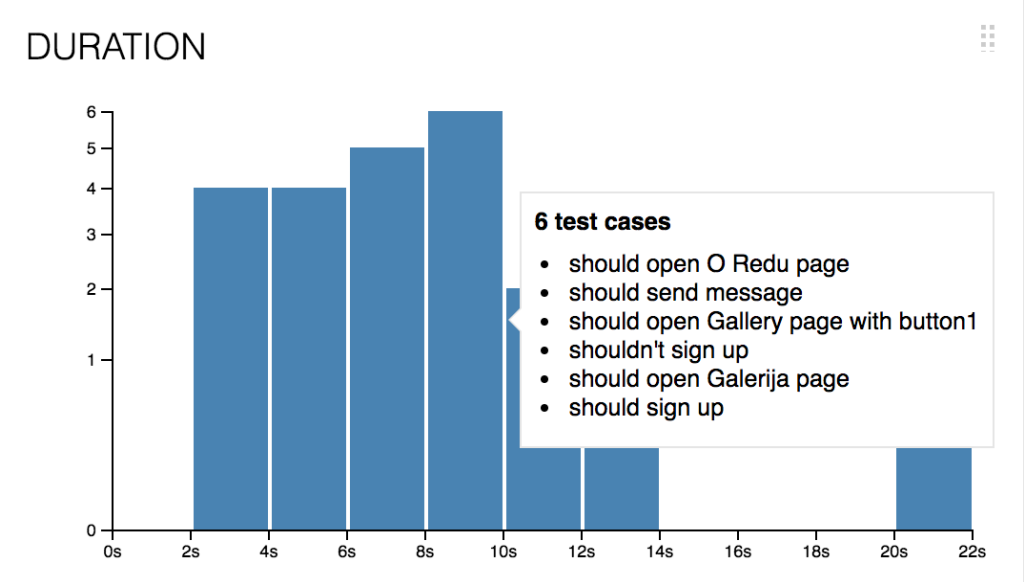
Kao što je prikazano na slici, definirano je vrijeme u kojem skripta čeka da se određeni element učita, no to se u ovom testnom scenariju nije dogodilo. Na slici je također prikazan identifikator elementa kojeg skripta nije mogla dohvatiti.

Na sljedećoj slici se nalazi grafički prikaz svih testnih scenarija koji su izvršeni. Grafički prikaz pod imenom "Status" prikazuje skup testnih scenarija koji su izvršeni, gdje zelena boja te postotak u sredini grafa pokazuju broj testnih scenarija koji su uspješno izvršeni, odnosno imaju pozitivan ishod. Isto tako crvena boja prikazuje broj testnih scenarija koji imaju negativan ishod, u ovom slučaju dva takva ishoda, koji predstavljaju testni slučaj gdje test provjerava da li se stranica galerije otvori nakon što korisnik klikne na gumb koji bi trebao voditi na istu. Žuta boja predstavlja broj testnih scenarija gdje skripta nije mogla locirati određeni element. Na slici 5.18 prikazan je grafički prikaz svih testnih scenarija koji su izvršeni u obliku testova nad novim web rješenjem udruge.



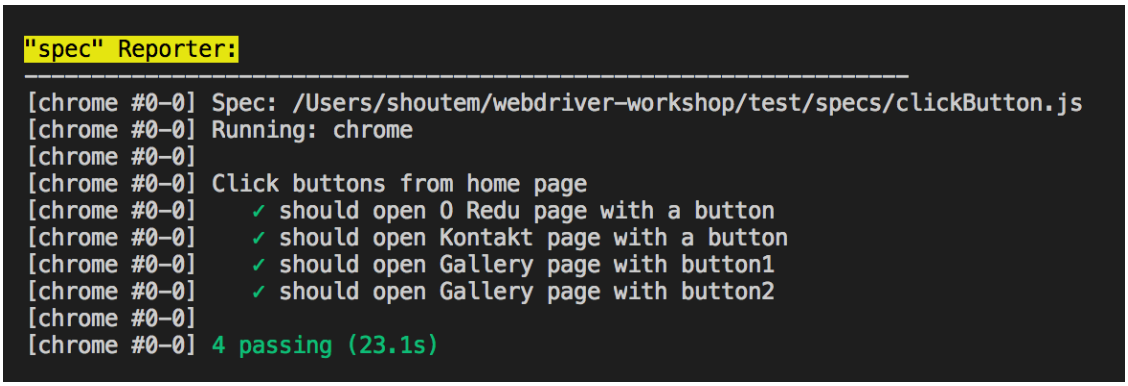
Slika 5.18 Prikaz prozora koji se sastoji od izvještaja generiranog alatom Allure

Na slici 5.18 također je vidljiva stavka pod nazivom *Duration* koja predstavlja vrijeme izvršavanja skripti, a izražena je u sekundama. Prikaz testnih scenarija jednog stupca prikazan je na slici 5.19.



Slika 5.19 Prikaz prozora koji se sastoji od izvještaja generiranog alatom Allure

Izvješća pojedinih testnih scenarija je moguće vidjeti i u samom terminalu, nakon izvršenja pojedine skripte. Na slici 5.20 je prikazan jedan takav izvještaj, gdje je skripta uspješno izvršena



Slika 5.20 Izvještaj generiran terminalom

Kao što je prikazano na slici, test ima pozitivan ishod, a u izvještaju je moguće vidjeti popis testnih scenarija koji su izvršeni u skripti pod nazivom *clickButton.js* koja ima zadatak provjere funkcionalnosti određenih elemenata. Također uz popis testnih scenarija, u izvještaju se nalazi naziv web-preglednika u kojem je testna skripta izvršena, te vrijeme koje je potrebno da se skripta izvrši.

Kao što agilni razvoj nalaže, održavanje digitalnog proizvoda predstavlja najveći dio razvoja, gdje je potrebno kontinuirano razvijati te testirati programsku podršku kako bi se održala što veća razina kvalitete samog proizvoda. S obzirom na to da je proizvod nad kojim je bilo izvršeno testiranje kvalitete u ovom slučaju novo web rješenje za udrugu "Red vitezova Ružice grada", proces testiranja i razvoja će se nastaviti prema zahtjevima klijenta. Dodavanjem novih značajki na web rješenje, povećava se potreba za izvršavanje novih testiranja, no s obzirom na to da je radno okruženje postavljeno, gdje su dostupni alati i za nadogradnju web rješenja i za testiranje novih elemenata i funkcionalnosti, to ne bi trebao biti problem jer je s daljnjim razvojem moguće nastaviti u bilo kojem trenutku. Svi ovi argumenti ukazuju da je agilni razvoj najbolje rješenje kod ovakve vrste proizvoda gdje uvijek postoji mjesto za napredak i poboljšanje samog web rješenja. Uz to, web rješenje je potrebno osvježavati s novim sadržajima, ne bi li to privuklo potencijalne članove da se pridruže udruzi u novim aktivnostima.

5.4.2 Dokumentacija testova

Sve testne slučajeve je potrebno pravilno prikazati, te prikladno analizirati. S obzirom na to da postoji veliki broj različitih testnih scenarija, testovi će biti podijeljeni na šest različitih grupa, gdje svaki pojedini testni plan predstavlja određenu funkcionalnost na web stranici. U tablici 5.1 prikazan je test koji provjerava otvaranje stranica koje se nalaze u glavnom izborniku.

Tablica 5.1 Test Otvranje stranica

	Otvaranje stranica	**Otvaranje stranica**	**Otvaranje stranica**	**Otvaranje stranica**
Web preglednik	Google chrome	Google chrome	Google chrome	Google chrome
Testni scenarij	Otvori stranicu 'O redu'	Otvori stranicu 'Ružica grad'	Otvori stranicu 'Galerija'	Otvori stranicu 'Članovi'
Koraci testa	1.Učitaj početnu stranicu 2.Dohvati element izbornika 'O redu' 3.Klikni na element 'O redu' 4.Pričekaj da se učita stranica 'O redu' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element izbornika 'Ružica grad' 3.Klikni na element 'Ružica grad' 4.Pričekaj da se učita stranica 'Ružica grad' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element izbornika 'Galerija' 3.Klikni na element ''Galerija'' 4.Pričekaj da se učita stranica ''Galerija'' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element izbornika 'Članovi' 3.Klikni na element ''Članovi'' 4.Pričekaj da se učita stranica ''Članovi'' i dohvati naslov 5.Usporedi URL
Testni podaci	oReduId const oReduUrl oReduTitle	ruzicaGradId ruzicaGradUrl ruzicaGradTitle	galerijaId galerijaUrl galerijaTitle	clanoviId clanoviUrl clanoviTitle
Prolaz/pad				

Kao što je i prikazano u tablici 5.1, test podrazumijeva otvaranje stranica koje se nalaze u glavnom izborniku. Prije svakog testnog scenarija učita se početna stranica, nakon čega testna skripta locira element u glavnom izborniku tako da dohvati identifikator, te klikne na isti taj element. Nakon klika na element u glavnom izborniku, skripta čeka da se učita sadržaj na stranici koju očekuje, te usporedi URL te stranice pomoću funkcije koja je dana u programskom kodu 5.23.

```
compareUrls(oReduUrl);
```

Programski kod 5.23 Funkcija za uspoređivanje URL-a

Funkcija koja uspoređuje URL stranice, dana je u programskom kodu 5.24.

```
function compareUrls(expectedUrl){
title = browser.getUrl();
assert.equal(title, expectedUrl); }
```

Programski kod 5.24 Funkcija compareUrls

Također je potrebno prikazati identifikatore koje je potrebno dohvatiti da bi se jedan testni scenarij otvaranja stranice uspješno izvršio, prema programskom kodu 5.25 dani su elementi potrebni za izvršavanje testnog scenarija otvaranja stranice pod nazivom O redu.

```
const oReduId = "#menu-item-289";
const oReduUrl = "https://vitezovi-ruzicegrada.com/o-redu/";
const oReduTitle = '//*[@id="oReduTitle"]/div/div/div/h1/span';
```

Programski kod 5.25 Definiranje elemenata O redu

U tablici 5.2 prikazan je test koji provjerava otvaranje stranica galerije klikom na gumbe koji se nalaze na početnoj stranici.

Tablica 5.2 Test "*clickButtonFail*"

	Klik na gumb	**Klik na gumb**
Web preglednik	Google chrome	Google chrome
Testni scenarij	Klikni na lijevi gumb	Klikni na desni gumb
Koraci testa	1.Učitaj početnu stranicu 2.Dohvati lijevi gumb 'Galerija' 3.Klikni na gumb 'Galerija' 4.Pričekaj da se učita stranica 'Galerija' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati desni gumb 'Galerija' 3.Klikni na gumb 'Galerija' 4.Pričekaj da se učita stranica 'Galerija' i dohvati naslov 5.Usporedi URL
Testni podaci	galerijaButton1Id galerijaButton2Id galerijaUrl	galerijaButton1Id galerijaButton2Id galerijaUrl
Prolaz/Pad		

Negativni testni scenarij koji je prikazan u tablici 5.2, također je prikazan prethodno pomoću alata Allure gdje skripta očekuje otvaranje stranice galerija, što se ne dogodi zbog neispravnog URL-a, a nakon korisničke radnje klika na gumb galerija koji se nalazi na početnoj stranici. Također je potrebno prikazati identifikatore koje je potrebno dohvatiti da bi se jedan testni scenarij klika na gumb galerije izvršio, prema programskom kodu 5.26 dani su elementi potrebni za izvršavanje testnog scenarija klika na gumb pod nazivom galerija.

```
it("should open page Galerija page with button1", function() {
$(galerijaButton1Id).waitForDisplayed();
$(galerijaButton1Id).click();
$(galerijaTitleFail).waitForDisplayed(); compareUrls(galerijaUrl); });
```

Programski kod 5.26 Testni scenarij klika na gumb galerija

U tablici 5.3 prikazan je test koji klikom na gumb otvara specifične stranice.

Tablica 5.3 Test *"*Klik na gumb*"*

	Klik na gumb	**Klik na gumb**	**Klik na gumb**	**Klik na gumb**
Web preglednik	Google chrome	Google chrome	Google chrome	Google chrome
Testni scenarij	Otvori stranicu 'O redu' klikom na gumb	Otvori stranicu 'Galerija' klikom na gumb	Otvori stranicu 'Galerija' klikom na gumb	Otvori stranicu 'Kontakt' klikom na gumb
Koraci testa	1.Učitaj početnu stranicu 2.Dohvati element gumba 'O redu' 3.Klikni na gumb 'O redu' 4.Pričekaj da se učita stranica 'O redu' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element lijevog gumba 'Galerija' 3.Klikni na gumb ''Galerija'' 4.Pričekaj da se učita stranica ''Galerija'' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element desnog gumba 'Galerija' 3.Klikni na gumb ''Galerija'' 4.Pričekaj da se učita stranica ''Galerija'' i dohvati naslov 5.Usporedi URL	1.Učitaj početnu stranicu 2.Dohvati element gumba 'Kontakt' 3.Klikni na gumb ''Kontakt'' 4.Pričekaj da se učita stranica ''Kontakt'' i dohvati naslov 5.Usporedi URL
Testni podaci	oReduButtonId const oReduUrl oReduTitle	galerijaButton1Id galerijaUrl galerijaTitle	galerijaButton2Id galerijaUrl galerijaTitle	kontaktButtonId kontaktUrl kontaktTitleId
Prolaz/pad				

Cilj testa koji je prikazan prema tablici 5.3 predstavlja testiranja svih gumba koji se nalaze na početnoj stranici. Potrebno je prikazati jedan testni scenarij, što je moguće učiniti na primjeru gumba pod nazivom O redu. Zajednička stvar svim scenarijima je ta da testna skripta nakon svakog testnog scenarija ponovno mora učitati početnu stranicu da bi mogla pristupiti sljedećem gumbu. U slučaju gumba O redu procedura bi bila sljedeća, gdje prvi korak predstavlja učitavanje početne stranice, nakon čega skripta mora dohvatiti gumb što čini pomoću naredbe koja je prikazana pomoću programskog koda 5.27.

```
$(oReduButtonId).waitForDisplayed();
```

Programski kod 5.27 Dohvaćanje ID-a gumba O redu

Varijabli *kontaktButtonId* je dodijeljen ID naziva *oReduButton*, koji je definiran preko alata Elementor, a dan prema programskom kodu 5.28.

```
const oReduButtonId = "#oReduButton";
```

Programski kod 5.28 Definiranje ID-a gumba O redu

Nakon dohvaćanja gumba, skripta nastavlja s izvođenjem, gdje je potrebno izvršiti još tri naredbe, koje su prikazane prema programskom kodu 5.29.

```
it("should open O Redu page with a button", function() {
..
$(oReduButtonId).click();
$(oReduTitle).waitForDisplayed();
compareUrls(oReduUrl); });
```

Programski kod 5.29 Klik na gumb i provjera valjanosti stranice

Nakon što je skripta dohvatila gumb, sljedeća radnja predstavlja klik na isti, nakon čega očekuje otvaranje stranice O redu, gdje dohvaća naslov na stranici, te isto tako uspoređuje URL stranice, pomoću funkcije *compareUrls.*

Tablica 5.4 prikazuje dokumentaciju testa za provjeru funkcionalnosti kontakt forme.

Tablica 5.4 Dokumentacija testnog scenarija *"messageForm"*

	Pošalji poruku	**Pošalji poruku**	**Pošalji poruku**
Web pretraživač	Google chrome	Google chrome	Google chrome
Testni scenarij	Poruka poslana	Invalid email	Prazna forma
Koraci testa	1.Učitaj kontakt stranicu i pričekaj sadržaj 2.Pronađi formu na stranici 3.Dohvati formu 4.Postavi ime i prezime 5.Postavi email 6.Postavi poruku 7.Pritisni gumb za slanje 8.Dohvati facebook button koji predstavlja sadržaj	1.Učitaj kontakt stranicu i pričekaj sadržaj 2.Pronađi formu na stranici 3.Dohvati formu 4.Postavi ime i prezime 5.Postavi krivi email format 6.Postavi poruku 7.Pritisni gumb za slanje 8.Dohvati facebook button koji predstavlja sadržaj	1.Učitaj kontakt stranicu i pričekaj sadržaj 2.Pronađi formu na stranici 3.Dohvati formu 4.Ostavi polje prazno 5.Postavi email 6.Postavi poruku 7.Pritisni gumb za slanje 8.Dohvati facebook button koji predstavlja sadržaj
Testni podaci	nameSurname Email Poruka faceButtonId #evf-submit-705 - gumb za slanje	nameSurname Email Poruka faceButtonId #evf-submit-705	nameSurname Email Poruka faceButtonId #evf-submit-705
Prolaz/pad			

U tablici 5.4 prikazan je test koji vrši testiranje forme za slanje poruke gdje prvi testni scenarij predstavlja pozitivan ishod jer su svi podaci uspješno upisani u formu. Preostala dva testna scenarija predstavljaju slučaj gdje je jedna od komponenti forme krivog formata.

Cilj poglavlja je prikazati rezultate testiranja na više načina, gdje je alat Allure korišten za vizualni prikaz izvješća nakon izvršenih testnih skripti putem terminala, a podaci koji se nalaze u tablicama prikazuju ručni unos podataka potrebnih za dokumentaciju pojedinih testnih scenarija. Potrebno je napomenuti da je u bilo kojem trenutku moguće napisati novi testni slučaj te ga pokrenuti putem terminala. Na taj je način moguće na vrlo brz te efikasan način provjeriti valjanost određenog elementa te njegove funkcionalnosti, ako se još u skriptu uključi i naredba za pozivanje alata Allure s ciljem da generira izvješće za taj pojedini testni slučaj, to je onda efikasan način testiranja s analizom rezultata u realnom vremenu.

6 ZAKLJUČAK

Potreba za novim digitalnim identitetom u obliku web rješenja može predstavljati dobru prilika za prikaz cjelokupnog agilnog razvojnog procesa programske podrške. Cilj rada je bio prikazati na koje je sve dijelove taj proces podijeljen te koliko je truda potrebno uložiti ne bi li se došlo do konačnog proizvoda, koji prema agilnom načelu nikad ne može biti u potpunosti biti dovršen s obzirom da je potrebno kontinuirano poboljšavati njegovu kvalitetu. Proces razvoja novog web rješenja počeo je s testiranjem starog web rješenja udruge, odnosno procesa korisničkog testiranja koje predstavlja odlično sredstvo za dobivanje uvida u korisničko iskustvo.

Prema dobivenim podacima korisničkog testiranja vidljivo je da staro web rješenje udruge ima dosta nedostataka koje su korisnici zamijetili. Glavni problemi i prepreke koje je potrebno naglasiti predstavljaju format teksta koji je iste tipografije na cijelom web rješenju, a na kojeg su korisnici imali primjedbe. Također, jedan od glavnih nedostataka na starom web rješenju je nedostatak URL strukture, gdje postoji samo jedna URL poveznica bez obzira na kojoj se stranici korisnik nalazio, a to otežava navigaciju kroz cijelo web rješenje. Iz tog je razloga nemoguće dohvatiti URL poveznicu pojedine stranice, što može biti frustrirajuće ako se želi ukazati na neki element ili dio stranice jednostavnim slanjem poveznice. Svi podaci testiranja koristili su se kod planiranja, a i samog razvoja novog web rješenja koje je razvijeno na platformi WordPress. Tijekom razvoja novog web rješenja također su provedeni automatizirani testovi pomoću alata Selenium WebDriver, a s ciljem postizanja što bolje kvalitete proizvoda, kao i s ciljem izbjegavanja većih pogrešaka. Isto tako, testiranje je potvrdilo ispravnost različitih funkcionalnosti koje su dodane na novo web rješenje, koje su dodane da bi olakšale navigaciju korisnika kroz web stranicu. Nakon postavljanja radnog okruženja, relativno je jednostavno napisati i pokrenuti nove testne planove, što je uz dodavanje novih funkcionalnosti na web rješenje ključ daljnjeg razvoja ovog i sličnih proizvoda.

LITERATURA

[1] I. Sommerville, Software Engineering, Pearson Education, Boston, 2011.

[2] Model vodopada, https://www.geeksforgeeks.org/software-engineering-classical-waterfall-model/, pristupljeno: lipanj 2019.

[3] Što je agilno?", https://www.agilealliance.org/agile101/, pristupljeno: lipanj 2019.

[4] Principi agilnog razvoja, https://sistemac.srce.hr/sites/sistemac.srce.hr/files/docs/seminari/2016/seminar-agilni-principi-razvoja-201609.pdf, pristupljeno: lipanj 2019.

[5] Tradicionalni proces razvoja web aplikacije, https://www.webfx.com/blog/web-design/agile/, pristupljeno: lipanj 2019.

[6] Agilni manifesto, https://agilemanifesto.org/, pristupljeno: lipanj 2019.

[7] User testing, https://www.everyinteraction.com/definition/user-testing/, pristupljeno: svibanj 2019.

[8] S. Portigal, Interviewing Users: How to Uncover Compelling Insights, Rosenfeld Media; 1st edition, 2013.

[9] Internet domena, https://sitebeginner.com/domains/what-is-a-domain-name/, pristupljeno: srpanj 2019.

[10] DNS, HTTP; https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works, pristupljeno: srpanj 2019.

[11] Web hosting; https://www.hostinger.com/tutorials/what-is-web-hosting/, pristupljeno: srpanj 2019.

[12] Pristup cPanel-u; https://www.hostinger.com/tutorials/what-is-cpanel, pristupljeno: srpanj 2019.

[13] WordPress; https://en.wikipedia.org/wiki/WordPress, pristupljeno: kolovoz 2019.

[14] K. Król, WordPress 4.x Complete, Fourth edition, Packt Publishing, 2015., str. 19-20, pristupljeno: kolovoz 2019.

[15] Prednosti i nedostaci platforme WordPress.org; https://www.wpbeginner.com/beginners-guide/self-hosted-wordpress-org-vs-free-wordpress-com-infograph/, pristupljeno: kolovoz 2019.

[16] Template; https://www.wpbeginner.com/glossary/template/, pristupljeno: kolovoz 2019.

[17] WordPress tema; https://wordpress.org/support/article/using-themes/, pristupljeno: srpanj 2019.

[18] K. Naik, P. Tripathy, Software Testing and Quality Assurance: Theory and Practice, 1st edition, John Wiley & Sons, 2008., pristupljeno kolovoz 2019.

[19] Kvaliteta programske podrške; http://softwaretestingfundamentals.com/software-quality/, pristupljeno: kolovoz 2019.

[20] Testiranje programske podrške; https://www.softwaretestingmaterial.com/software-testing/, pristupljeno kolovoz 2019.

[21] J. Rasmusson, The Way of the Web Tester A Beginner's Guide to Automating Tests, Pragmatic Bookshelf, 2016., str: 5-30, pristupljeno kolovoz 2019.

[22] Selenium; https://www.seleniumhq.org/docs/01_introducing_selenium.jsp, pristupljeno: kolovoz 2019.

[23] Same origin-policy; https://en.wikipedia.org/wiki/Same-origin_policy, pristupljeno: kolovoz 2019.

[24] Selenium; Selenium; https://www.guru99.com/introduction-to-selenium.html, pristupljeno: kolovoz 2019.

[25] U. Gundecha, Selenium WebDriver 3 Practical Guide - Second Edition, Packt Publishing, 2018., pristupljeno: kolovoz 2019.

[26] JavaScript; https://en.wikipedia.org/wiki/JavaScript, pristupljeno: kolovoz 2019.

[27] cPanel značajke; https://documentation.cpanel.net/display/82Docs/cPanel+Features+List, pristupljeno: kolovoz 2019.

[28] Osnovne informacije cPanel računa; https://documentation.cpanel.net/display/82Docs/The+cPanel+Interface#ThecPanelInterface-general, pristupljeno: kolovoz 2019.

[29] Primary and Secondary Nameserver; https://www.cloudflare.com/learning/dns/glossary/primary-secondary-dns/, pristupljeno: kolovoz 2019.

[30] HTTPS; https://www.cloudflare.com/learning/ssl/what-is-https/, pristupljeno: kolovoz 2019.

[31] Defaultni izgled WordPress web stranice; https://wordpress.org/support/article/first-steps-with-wordpress-classic/, pristupljeno: kolovoz 2019.

[32] curl; https://dev.to/ibmdeveloper/what-is-curl-and-why-is-it-all-over-api-docs-9mh, pristupljeno: rujan 2019.

[33] WebDriver; https://www.seleniumhq.org/docs/03_WebDriver.jsp, pristupljeno: rujan 2019.

SAŽETAK

Svrha ovog diplomskog rada je agilnim pristupom u razvoju programske podrške izraditi novo web rješenje jedne udruge. Pri tome je provedeno korisničko testiranje postojećeg web rješenja na određenom broju različitih korisnika, te prema rezultatima testiranja planiran postupak razvoja novog web rješenja. U to rješenje implementirani su svi elementi i funkcionalnosti prema prijedlozima korisnika, te prema zahtjevima vlasnika udruge. Tijekom razvoja web rješenja, obavljen je niz testova s ciljem povećanja kvalitete programskog proizvoda. Izvedeno je automatizirano testiranje korisničkog sučelja, odnosno svih funkcionalnosti koje se nalaze na istom, a automatiziranim testiranjem omogućeno pravilno funkcioniranje istih. Korisničko testiranje je provedeno kroz niz razgovora s korisnicima, a na temelju prikupljenih podataka razvijeno je novo web rješenje koristeći platformu WordPress koja pruža brz i pouzdan razvoj web stranica. Isto tako, u bilo kojem trenutku se moglo dodavati nove funkcionalnosti i prikladno ih testirati, pri čemu je važnu ulogu imao alat Elementor, koji omogućuje dodavanje novih elemenata iz samog sučelja.

Ključne riječi: agilni razvoj, automatizirano testiranje, korisničko testiranje, Selenium, web aplikacija, WordPress.

ABSTRACT

The purpose of this thesis is to develop a new web solution for an association through an agile approach in software development. In doing so, user testing of the existing web solution was carried out on a number of different users, and according to the test results, a planned process of developing a new web solution was planned. All elements and functionalities are implemented in this solution according to the suggestions of the users and according to the requirements of the owners of the association. During the development of the web solution, a series of tests were carried out in order to increase the quality of the software product. Automated testing of the user interface, that is, all the functionalities on the same was performed, and automated testing enabled the proper functioning of the web solution. User testing was conducted through a series of conversations with users, and based on the data collected, a new web solution was developed using the WordPress platform, which provides fast and reliable web development. Also, new functionalities could be added and tested at any time, with the Elementor tool playing an important role, allowing new elements to be added from the interface itself.

Keywords: agile development, automated testing, user testing, Selenium, web application, WordPress.

ŽIVOTOPIS

Ivan Balogović rođen je 18.7.1991. godine u Koprivnici. Završio je strukovnu školu u Đurđevcu - smjer Tehničar za računalstvo. Akademske godine 2010./2011. upisao je stručni studij Informatike na Elektrotehničkom fakultetu u Osijeku, današnjem Fakultetu elektrotehnike, računarstva i informacijskih tehnologija. Nakon završenog stručnog studija upisuje Razlikovne obveze s ciljem nastavljanja studiranja na Diplomskom studiju računarstva. Početkom 2018. godine počinje raditi kao student u tvrtki FIVE, gdje i trenutno radi. A od kraja 2018. godine zaposlenik je tvrtke FIVE.

PRILOZI

Prilog 1. Pismeni oblik rada u .doc i .pdf formatu

Prilog 2. Programski projekt web aplikacije

Prilog 2. Programski kodovi provedenih testova u obliku projekta alata Visual Studio Code