Izrada portala za Studentski zbor i studente FERIT-a

Tominović, Matea

Master's thesis / Diplomski rad

2018

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:200:928701

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Diplomski studij

PORTAL ZA STUDENTSKI ZBOR I STUDENTE FERITA

Diplomski rad

Matea Tominović

Osijek, 2018.

SADRŽAJ:

1. UVOD 1

1.1. Zadatak završnog rada 1

2. OPĆENITO O WEB STRANICAMA 2

2.1. Definicija web stranice 2

2.2. Internetski preglednici 2

2.3. Glavni elementi web stranica 3

2.4. Izrada web stranice 3

3. PREGLED KORIŠTENIH TEHNOLOGIJA 4

3.1. HTML 4

3.2. CSS 4

3.3. PHP 5

3.4. jQuery 5

3.5. JavaScript 5

3.6. Bootstrap 6

3.7. Laravel 6

3.8. Baza podataka i MySQL 7

4. SASTAVNICE WEB STRANICE STUDENTSKOG ZBORA 9

4.1. Podstranice 9

4.1.1. Naslovna stranica 9

4.1.2. Fakultet 10

4.1.3. Demonstrature 12

4.1.4. Studenti 13

4.1.5. Elektrijada 14

4.1.6. Studentska putovanja 15

4.2. Dashboard 16

5. FUNKCIONALNOST STRANICE STUDENTSKOG ZBORA 18

5.1. Studentski zbor kao registrirani korisnik 19

5.2. Prodekan za nastavu i studente kao registrirani korisnik 27

5.3. Student kao registrirani korisnik 28

6. ZAKLJUČAK 31

LITERATURA 32

SAŽETAK 33

ABSTRACT 34

ŽIVOTOPIS 35

1. UVOD

Web aplikacije predstavljaju programska rješenja, dizajnirana u nekom od programskih jezika, kojima se može pristupiti putem različitih internet preglednika služeći se Internetom. Dostupnost web aplikacija u bilo koje vrijeme, bilo to s mobilnog uređaja ili računala, predstavlja glavnu prednost takvih aplikacije te ujedno potiču njihov nagli rast i razvoj. U današnje vrijeme gotovo sva poduzeća, ustanove, kao i oglasnici imaju vlastite web stranice koje su postale apsolutni „must have“. Web stranice idealno su mjesto za promidžbu tvrtke, ustanove ili pojedinca, a upravo one komuniciraju umjesto nas sa ostatkom svijeta, to jest s mnoštvom korisnika Interneta koji su web stranicu posjetili planski ili slučajno. One mogu biti odraz nas ili našeg poslovanja, ovisno o tome za što je web stranica namijenjena. Različite su vrste web stranica, ovisno o namjeni, strukturi i izradi, a najčešće za svrhu imaju prezentirati vlasnika ili njegovu djelatnost.

Studentski zbor predstavlja zakonom određeno studentsko izborno predstavničko tijelo čija je glavna uloga zaštita interesa studenta te predstavljanje studenata u sustavu visokog obrazovanja. Postoji zbog studenata, kako bi im pomogao pri ostvarivanju njihovih prava.

Kao član Studentskog zbora uvidjela sam potrebu za izradom web stranice koja bi Studentskom zboru poslužila kao mjesto oglašavanja i jednostavnijeg komuniciranja sa studentima, ali i poslužila studentima kao mjesto gdje bi lakše potražili pomoć vezano uz ispite, pronalaženje cimera ili se prijavili za određeno studentsko putovanje ili natjecanje.

Web stranica napravljena je koristeći Laravel open-source PHP razvojno okruženje (*engl. framework*). Od razvojnih tehnologija osim Laravela korišteni su: HTML, CSS, PHP, jQueri, Java Script, Bootstrap i MySQL.

1.1. Zadatak završnog rada

Zadatak završnog rada je izrada portala za Studentski zbor i studente FERIT-a.
Omogućiti prijavu studenata na portal sa njihovim ovlastima, te prijavu za administratore i urednike portala. Studentski zbor može postavljati obavijesti za studente, otvarati različite natječaje, prijave za Elekrijadu, objavljivati oglase za poslove i slično. Također, studenti se mogu registrirati i postavljati vlastite obavijesti po kategorijama: npr. potraga za cimerom/smještajem, pomoć za određeni kolegij, instrukcije i slično. Za izradu rada koristiti će se Laravel razvojno okruženje.

2. OPĆENITO O WEB STRANICAMA

World Wide Web (skraćeno WWW, W3, ili samo Web) jedna je od najkorištenijih usluga Interneta koja služi za dohvaćanje hipertekstualnih dokumenata. Sadržaji tih dokumenata mogu biti različitog tipa, kao što su tekst, slike i multimedijski sadržaji, koji su međusobno su povezani tzv. hiperlinkovima. Za dohvaćanje i prikaz sadržaja takvih dokumenata koriste se Web preglednici.

2.1. Definicija web stranice

Web stranica predstavlja web dokument pogodan za World Wide Web i internetski preglednik koji služi za prikaz web stranice na računalu ili mobitelima. Web stranica je ono što se prikazuje, ali termin označava i datoteku čija je glavna funkcija obezbijediti hypertext koji će navigirati prema ostalim web stranicama preko linkova [1].

Web stranica je HTML dokument koji omogućava prezentaciju teksta i multimedijskih elemenata kao što su slike u digitalnom formatu, različite animacije, video i zvuk.

2.2. Internetski preglednici

Internetski preglednik *(engl. Browser)* je program koji se koristi za pregled web stranica na Internetu, kao i na lokalnom serveru, a radi na način da učita HTML dokumente sa web stranice i prikaže ih na monitoru [2]. Glavne osobine koje treba posjedovati svaki internetski preglednik su: sigurnost, brzina, mogućnosti, jednostavnost korištenja i dokumentacija za podršku i pomoć.

Svi glavni internetski preglednici omogućavaju korisniku otvaranje više različitih informacijskih resursa u isto vrijeme, bilo to u više odvojenih prozora preglednika ili u različitim karticama u istom prozoru. Također, glavni preglednici omogućavaju korisniku blokiranje skočnih prozora kako bi se spriječilo iskakanja neželjenih prozora bez korisnikova odobrenja.

2.3. Glavni elementi web stranica

Web stranica, kao skup informacija, može sadržavati različite vrste informacija koje korisnici mogu vidjeti, čuti ili koje mogu ostati skrivene prema korisniku [1].
Dva osnovna tipa elemenata web-stranica su:

1. Uočene (renderirane) informacije:
    - Tekstualne informacije
    - Ne-tekstualne informacije: statične slike, animacije, zvučne informacije ili video
    - Interaktivne informacije: Za interakciju na stranici ili između stranica
2. Unutrašnje (skrivene) informacije:
    - Komentari
    - Povezane datoteke preko hiperlinkova
    - Dijagrami i informacije o stilu
    - Skripte

2.4. Izrada web stranice

Programeri i web dizajneri bave se izradom modernih web stranica.
Kao i sve druge složene aplikaciju, web stranice čine sljedeće razine: [2]

- Podatkovna razina (tekst, podaci i podatkovni sadržaj)
- Poslovna ili interakcijska razina (program, skripta)
- Prezentacijska razina (dizajn)

Iz navedenih osobina, zaključujemo da je za izradu moderne web stranice potrebno podjednako dobro poznavati informatičku tehnologiju kao i imati smisla za dizajn. Uspješni web dizajner trebao bi posjedovati osobine inženjera, programera kao i umjetnika.Na izradi ozbiljnijih i velikih web stranica može sudjelovati veći broj stručnjaka zaduženih za određeni dio web stranice pa tako zajedno mogu raditi programeri, grafički dizajneri, novinari, fotografi kao i stručnjaci za marketing s ciljem postizanja što boljeg konačnog proizvoda.

3. PREGLED KORIŠTENIH TEHNOLOGIJA

3.1. HTML

HTML je prezentacijski jezik koji se koristi za izradu web stranica. Svojim oznakama HTML definira web pregledniku kako rasporediti sadržaj web stranice i kako stranica treba izgledati u konačnici. Poznavanje HTML jezika neophodno je u web dizajnu i predstavlja ono od čega svaki programer treba krenuti.

HTML nije programski, već opisni jezik koji služi za opis hipertekstualnih dokumenata. Njime nismo u mogućnosti izvršiti nikakav zadatak pa čak ni najjednostavnije računalne operacije, kao ni provjeravati uvjete ili vrtjeti petlje. HTML datoteke obične su tekstualne datoteke, a ekstenzija im je .html ili .htm [3].

3.2. CSS

U početku su unutar HTML-a ubačeni elementi za definiciju prezentacije (npr. tag <font>), ali s razvojem weba brzo je uočena potreba za stilskim jezikom čija je glavna uloga osloboditi HTML od potrebe prikazivanja i oblikovanja sadržaja. Upravo tome danas služi CSS.

CSS je stilski jezik koji definira kako prikazati HTML elemente pa se njime uređuje sam izgled i stil web stranice [4].

CSS kod se sastoji od tri glavna elementa: [4]

- Selektori – identificiraju određene elemente na HTML stranici i utječu na njihov izgled
- Svojstva – služe za opis pojedinih svojstva kao što su boja teksta, veličina teksta itd.
- Vrijednosti – predstavljaju vrijednost koju može poprimiti pojedino svojstvo

Sintaksa CSS-a definirana je na slijedeći način:

selektor { svojstvo: vrijednost; }

CSS kod moguće je povezati sa HTML dokumentom na slijedeće načine:

- Internim povezivanjem,
- Vanjskim povezivanjem,
- Umetanjem CSS svojstava unutar samog HTML elementa.

3.3. PHP

PHP je programski jezik koji je dizajniran za izradu dinamičkih i interaktivnih web stranica, a ujedno je jedan od najpopularnijih i najnaprednijih skriptnih jezika današnjice. PHP skripte izvode se na serveru (engl. server side) umjesto na klijentskom računalu i služe za dinamičko generiranje HTML koda [5].

PHP sadrži funkcije kojima može manipulirati velikim brojem različitih tipova sadržaja kao što je manipulacija grafikom, loadanje .NET modula i rad sa XML-om.

Glavna prednost PHP-a ispred ostalih web skriptnih tehnologija je njegova podrška za korištenjem različitih tipova baza podataka kao što su: MySQL, PostgreSQL, dBase, Oracle i druge.

PHP skripni jezik je besplatan i otvorenog koda, a zbog svojih dobrih osobina najčešće postaje prvi izbor za izradu vlastitih mrežnih sustava velikih i malih kompanija.

3.4. jQuery

jQuiery je biblioteka koja omogućava brzu i jednostavnu izradu web stranica. Služeći se jQuery-jem pomoću jedne linije koda možemo postići isto što bi se postiglo sa desetak linija regularnog Java Script koda [6].

Najveća prednost jQuery-ja vidljiva je prilikom:

- Dodavanja efekata tekstu,
- Izradi slide showa,
- Kreiranja drag&drop sučeljaa,
- Izradi formulara.

JQuery besplatan je za skidanje i uporabu, a kompatibilan je s brojnim pretraživačima.

3.5. JavaScript

JavaScript je skriptni programski jezik koji se, za razliku od PHP-a, izvršava na strani korisnika, tj. izvodi se na klijentskom osobnom računalu (*engl. client side*). Napravljen je s ciljem sličnosti Javi, ali nije objektno orijentiran kao Java već se temelji na prototipu.

Osnovna svrha JavaScripta je to što ovaj skriptni jezik omogućava interaktivnost između više različitih elemenata stranice. JavaScript se također koristi i za poboljšanje funkcionalnosti stranice, validaciju formi, asinkronu komunikaciju sa serverom i još mnogo toga [7].

Na tri načina možemo povezati JavaScript s HTML kodom:

- Spremiti ga u tijelo stranice (*engl. Body*)
- Spremiti ga u zaglavlje stranice (*engl. Head*)
- Spremiti ga u posebnu JavaScript datoteku sa ekstenzijom (.js)

JavaScript posjeduje relativno mali broj tipova podataka (numeričke, boolean i string vrijednosti), a podržava i funkcije bez posebnih zahtjeva za deklariranjem.

JavaScript jedan je od najpopularnijih skriptnih jezika na Internetu, a podržavaju ga svi poznatiji internetski preglednici.

3.6. Bootstrap

Bootstrap predstavlja radni okvir koji je besplatan i otvorenog koda. Osmišljen s ciljem bržeg i lakšeg razvoja web stranica. Kombinacija je HTML-a, CSS-a i JavaScript-a, razvijen sa ciljem olakšanog razvoja web formi kao i razvoja naprednih web komponenti. Bootstrap predstavlja kolekciju razvijenih CSS i JavaScript alata i biblioteka [8].

Pomoću Bootstrap-a lako stvaramo odgovarajući dizajn stranice koji se izgledom može dobro prilagoditi svim uređajima na kojima se prikazuje, bio to mali telefon i veliki desktop.

Bootstrap olakšava integriranje raznih vrsta komponenti kao što su forme, button-i, akcija sa tekstom, i slično, a isto tako jako dobro surađuje sa JavaScript-om i sa jQuery bibliotekama.

Glavni nedostatak Bootstrapa je što web stranicama ne daje orginalni izgled, već većina aplikacija, u kojima se Bootrstap koristi, izgledom podsjećaju jedna na drugu.

3.7. Laravel

Laravel je open-source PHP razvojno okruženje (*engl. framework*). Namijenjen je za razvoj web aplikacija upotrebom MVC (*engl. Model-View-Controler*) arhitekture. Smatra se jednim od najpoznatijih PHP frameworka na svijetu [9].

Glavne prednosti Laravela su sljedeće:

- Jednostavna i izražajna sintaksa,
- Dizajniran za jednog programera ili čitavi tim,
- Omogućava sinkronizaciju koristeći migracije baze podataka
- Koristi moderne alatke kao što su: ORM, rutiranje, jednostavna autentifikacija i drugi,
- Omogućava jednostavno održavanje stranice.

Kako bi se Laravel mogao koristiti na korisničkom računalu potrebno je instralirati Composer koji korisnicima omogućava upravljanje komponentama od kojih se sastoji Laravel framework. Dobra stvar je ta što se prilikom instaliranja Composer-a vrši automatska nadogradnja njegove najnovije verzije.

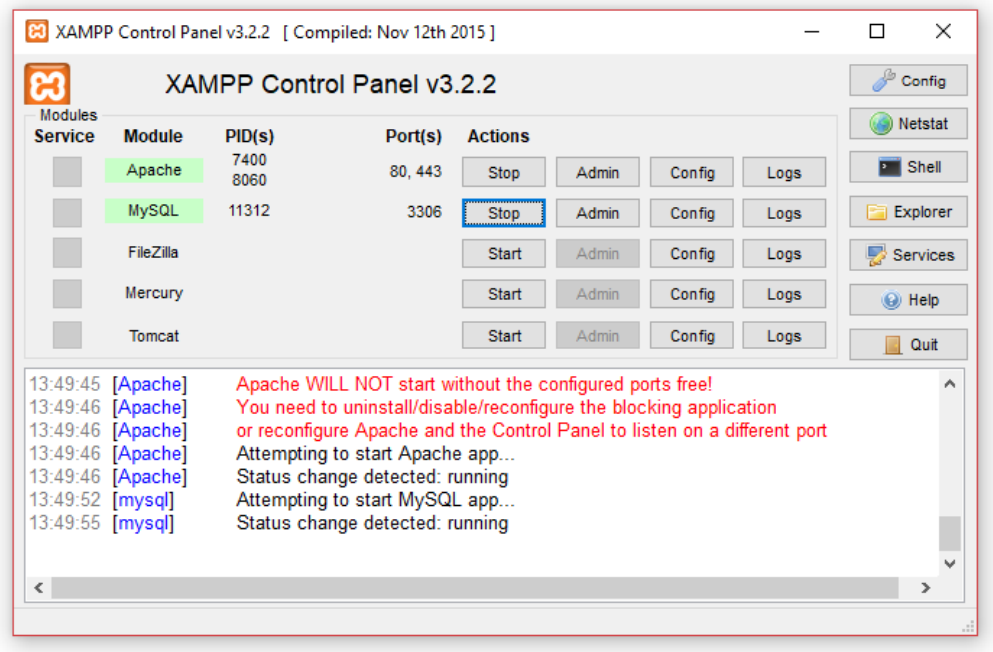
3.8. Baza podataka i MySQL

MySQL predstavlja sistem za upravljanje relacijskim bazama podataka. Svakoj MySQL bazi može pristupiti nekoliko korisnika, a svaki korisnik ima definirana ovlaštenja i mogućnosti za rad s tom bazom.

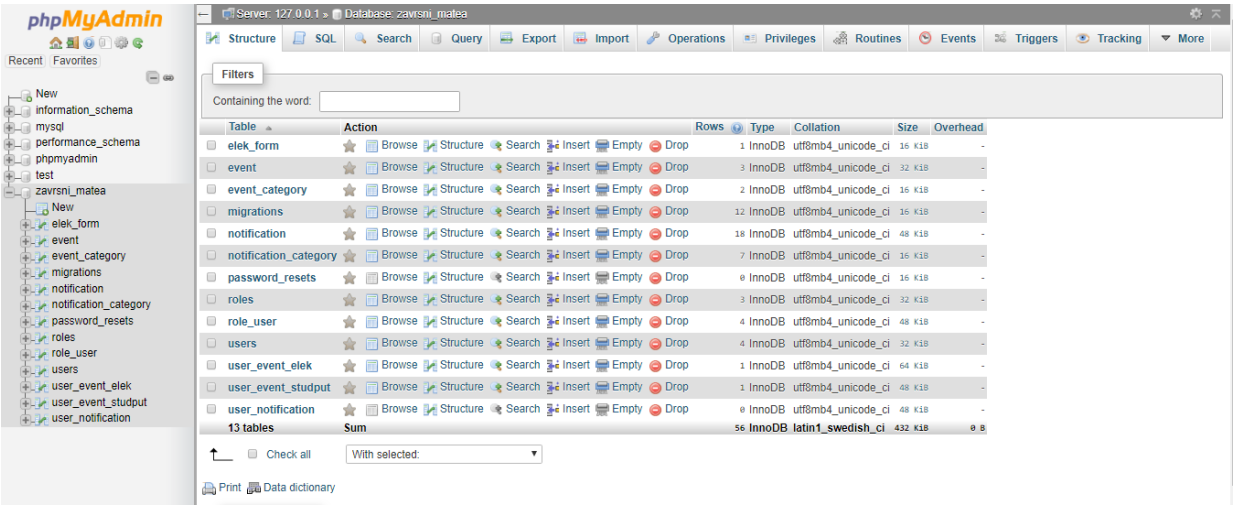
Server može sadržavati veći broj različitih baza podataka koje su potpuno samostalne, ali unutar jednog projekta moguće je služiti se podacima iz više različitih baza na serveru. Svakom korisničkom računu na serveru je moguće dodijeliti različita administrativna prava koja se odnose na cijeli server ili pak na pojedinačne baze. Neka od prava odnose se na prava pristupa već postojećim bazama podataka, uređivanje postojećih baza, kao i stvaranje novih baza.

Za realizaciju aplikacije potrebno je kreirati bazu podataka koja sadrži tablice u kojima su smješteni podatci o postavljenim obavijestima, registriranim korisnicima na portalu i slično. Za tu primjenu koristi se program XAMPP, koji je besplatan program otvorenog koda i omogućava korisniku kreiranje lokalne baze na svom računalu.

Nakon što pokrenemo program XAMPP, potrebno je pokrenuti *(engl. Start)* Apache i MySql servere kako bi bilo moguće pristupiti kontrolnoj ploči baze podataka. Kontrolnoj ploči pristupamo na način da u adresu Internet pretraživača upišemo *localhost/phpmyadmin*.



Slika 3.1. *Izgled kontrolne ploče XAMPP-a*



Slika 3.2. *Kontrolna ploča baze podataka*

4. SASTAVNICE WEB STRANICE STUDENTSKOG ZBORA

Portal za Studentski zbor i studente FERIT-a zamišljen je kao mjesto jednostavnog oglašavanja i komunikacije između studenata fakulteta. Studentski zbor kao predstavničko tijelo studenata učestalo organizira različita gostujuća predavanja, radionice, demonstrature, studentska putovanja ili natjecanja. Upravo je ovaj portal zamišljen kao mjesto na kojem će studenti moći saznati sve vezano uz aktivnosti koje za njih organizira Studentski zbor, ali i Fakultet. Studenti će pod različitim kategorijama moći potražiti pomoć vezano uz ispite na fakultetu. Također, portal će im omogućiti jednostavnije pronalaženje budućih cimera, a na portalu će se moći i prijaviti za studentska putovanja i natjecanja koja organizira Studentski zbor.

4.1. Podstranice

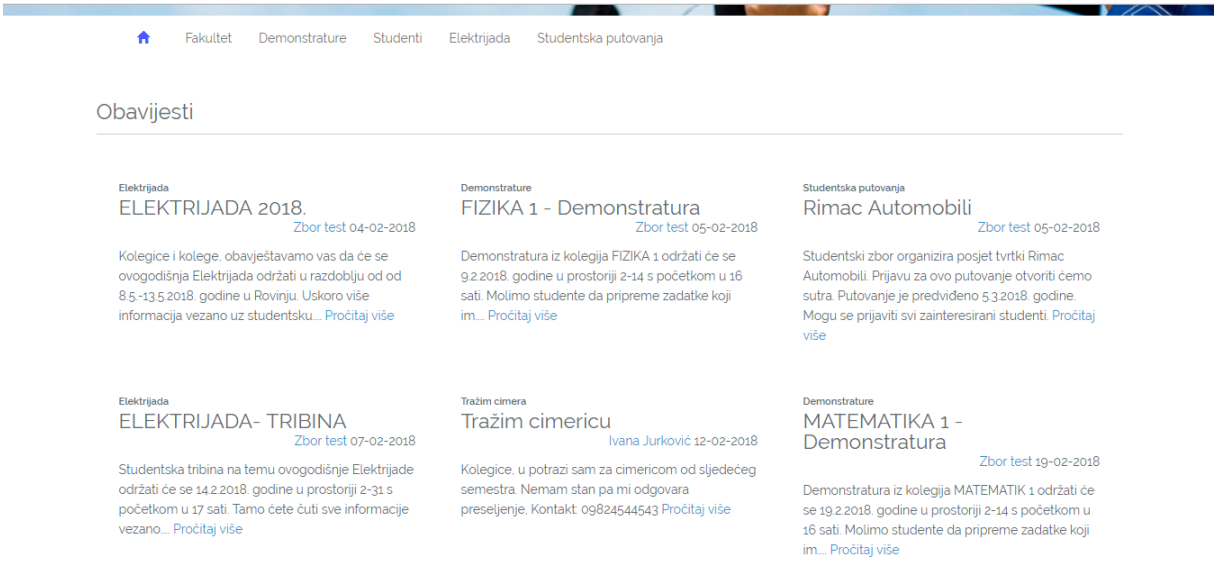
Portal Studentskog zbora sastoji se od 6 podstanica: *Naslovna stranica, Fakultet, Demonstrature, Studenti, Elektrijada* i *Studentska putovanja*. Na svakoj podstranici definirana je njezina uloga i presjek posljednje tri obavijesti, dok *Naslovna stranica* prikazuje posljednjih 6 objavljenih obavijesti iz bilo koje kategorije. Na podstranicama *Elektrijada* i *Studentska putovanja*, osim obavijesti, mogu se objavljivati eventi, tj natječaju za prijavu studenata.

4.1.1. Naslovna stranica

Naslovna stranica, koja predstavlja prvu podstranicu portala Studentskog zbora za studente FERIT-a, prvo je što studenti vide kada posjete navedeni portal. Naslovna stranica prikazuje 6 posljednje objavljenih oglasa iz bilo koje kategorije, od najnovije objavljenje prema najstarijoj. Gost portala u zaglavlju ima mogućnost *Login-a* ili *Registracije*. Za svaku obavijest na naslovnoj stranici definirana je kategorija kojoj pripada, naslov, kreator obavijesti, datum obavijesti i sadržaj. Ukoliko je obavijest dužeg sadržaja, pritiskom na *„Pročitaj više“* korisniku se otvara obavijest sa kompletnim sadržajem.



Slika 4.1. *Izgled Naslovne (Home) stranice*

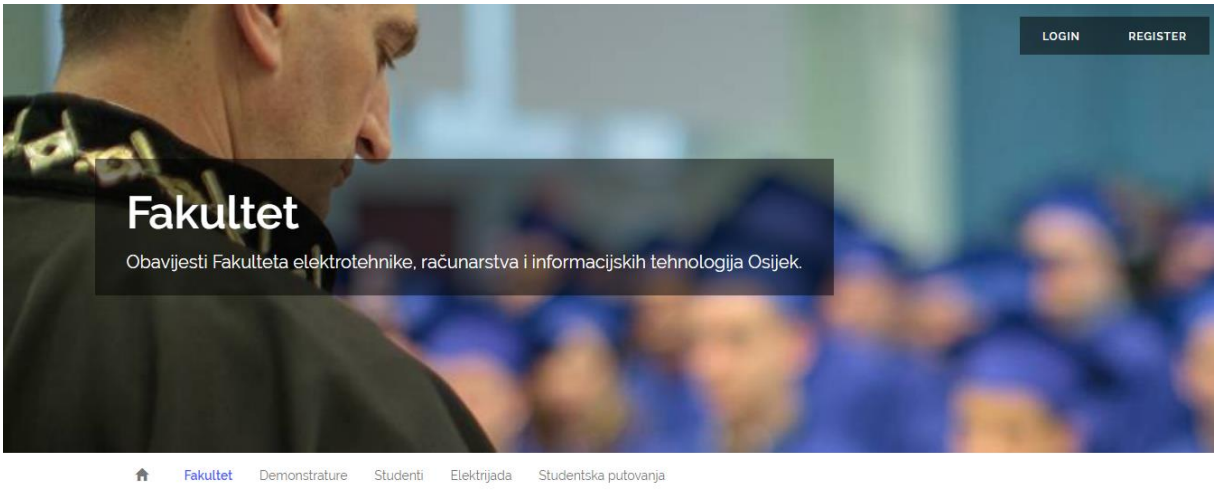


Slika 4.2. *Prikaz obavijesti na Naslovnoj stranici*

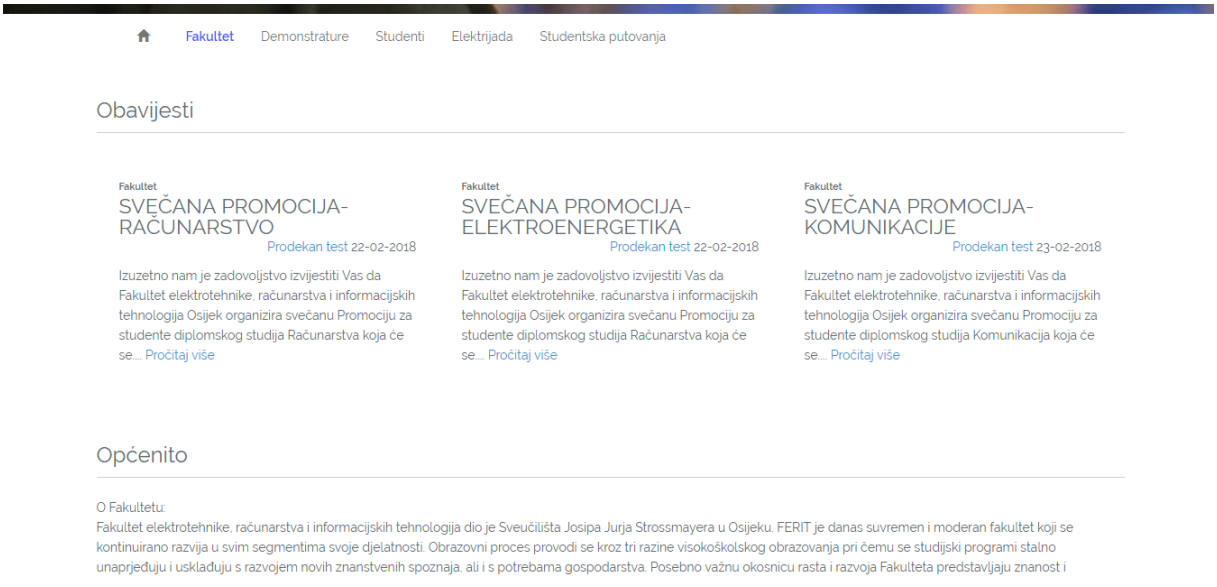
4.1.2. Fakultet

Fakultet je podstanica portala na kojoj studenti mogu vidjeti osnovne informacije o Fakultetu, kao što su ustroj fakulteta, zavode fakulteta, upravu, ali lokaciju na kojoj se nalazi putem Google mape. Također, na ovoj podstranici studenti mogu vidjeti obavijesti koje su vezane uz Fakultet, kao što su obavijesti vezane za održavanje svečanih promocija, natječaji za zakup prostorija na Fakultetu, natječaji za nova radna mjesta i slično. Obavijesti u ovoj kategoriji

može postavljati Prodekan fakulteta ili Studentski zbor. Studenti na ovoj podstranici mogu vidjeti 3 posljednje objavljene obavijesti iz ove kategorije, od najnovije prema najstarijoj.



Slika 4.3. *Izgled podstanice Fakultet*

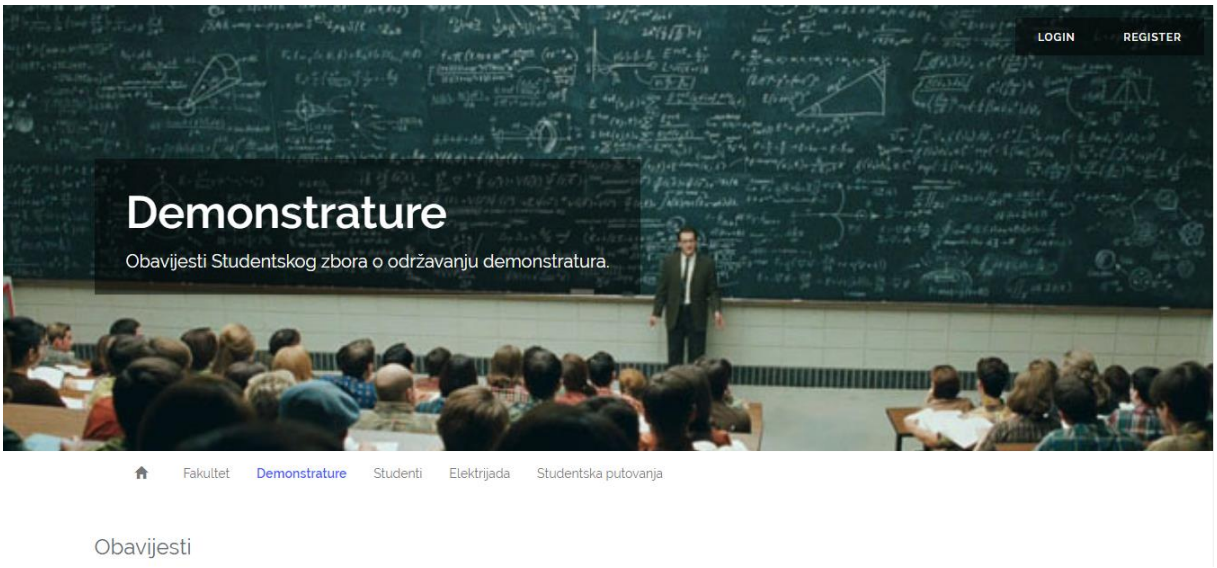


Slika 4.4. *Pregled obavijesti na podstranici Fakultet*

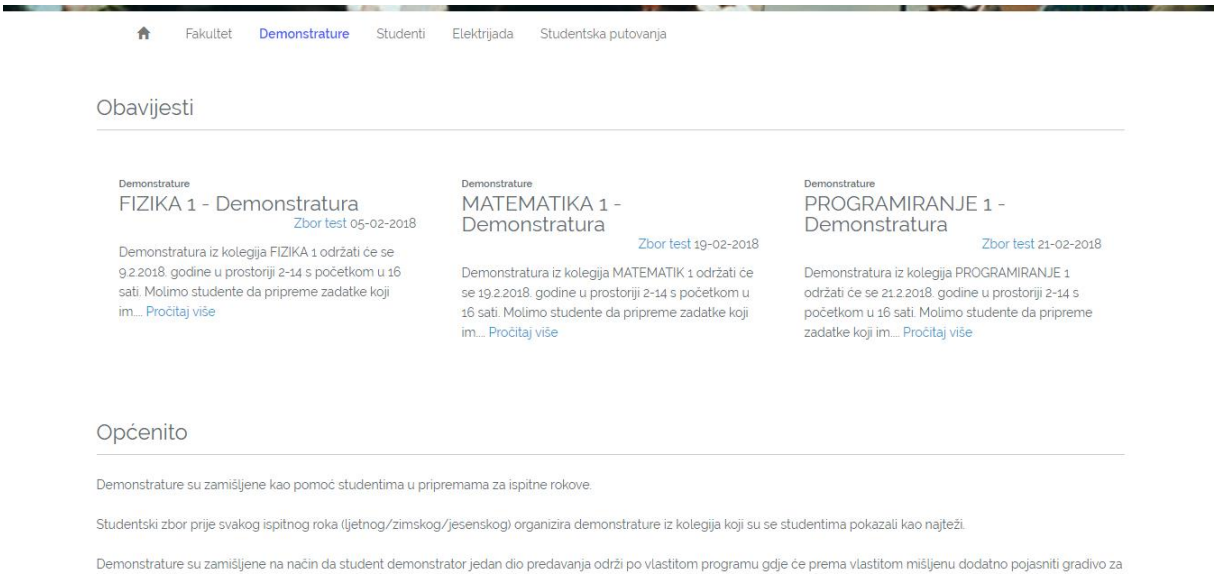
4.1.3. Demonstrature

Studentski zbor, svake godine prije ispitnih rokova, organizira demonstrature iz kolegija koji su se studentima pokazali kao najteži za svladavanje. Demonstrature su zamišljene kao grupne konzultacije na kojima student, kojemu određeni predmet „ide od ruke“, objašnjava svojim kolegama te im olakšava polaganje istog na nadolazećim ispitnim rokovima.

Iz tog razloga dodana je ova podstranica na kojoj će studenti imati pregled obavijesti o održavanju demonstratura na Fakultetu.



Slika 4.5. *Izgled podstranice Demonstrature*



Slika 4.6. *Pregled obavijesti na podstranici Demonstrature*

4.1.4. Studenti

Podstranica *Studenti* sadrži tri kategorije obavijesti. Studenti mogu postavljati obavijesti pod kategorijama: *Tražim cimera* i *Tražim/dajem instrukcije,* dok Studentski zbor može objavljivati pod kategorijom *Izvannastavne aktivnosti.*

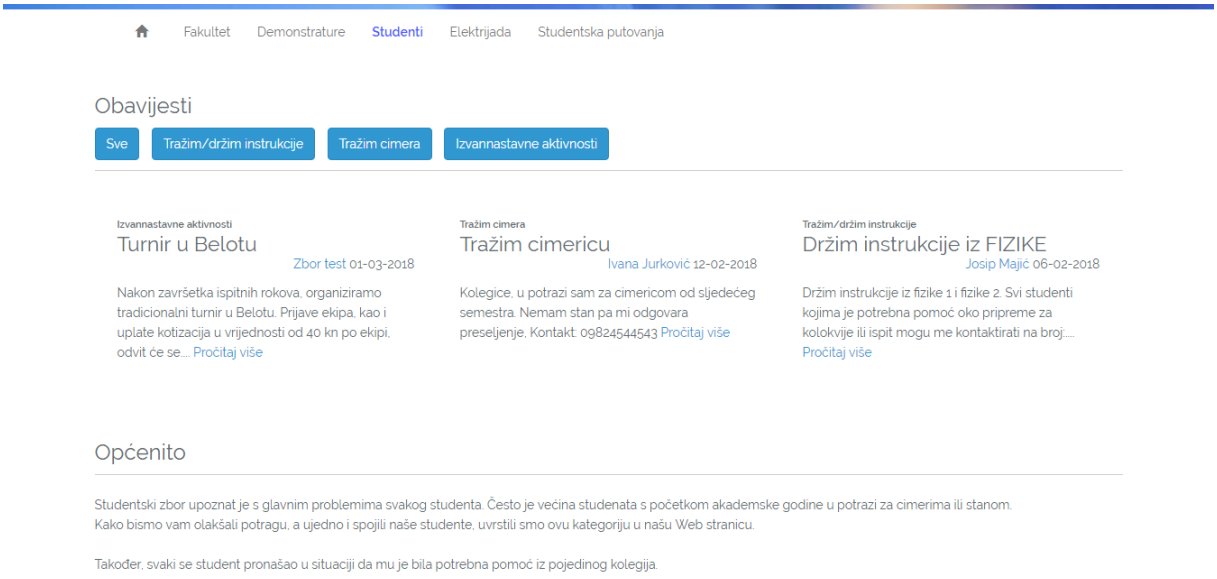
Svaki se student ponekad našao u situaciji da mu je bila potreba pomoć oko pojedinog kolegija, a upravo ova podstanica može mu olakšati pronalaženje adekvatnog instruktora. Studenti mogu postavljati oglase da održavaju instrukcije svojim kolegama te ih tako pripremaju za nadolazeće ispite, ili student kojemu je potrebna pomoć objavom oglasa može lakše pronaći studenta koji bi mu pomogao.

Također, većina studenta početkom akademske godine nastoji pronaći cimera ili stan, a postavljanjem obavijesti na ovoj podstranici student si može olakšati i ubrzati potragu.

Budući da Studentski zbor često organizira druženja studenata izvan okvira nastave, kao što su turniri u Belotu, turniri u računalnim igricama, pub kvizovi i sl., upravo će obavijesti vezane za takva druženja objavljivati pod kategorijom Izvannastavne aktivnosti.



Slika 4.7. *Izgled podstranice Studenti*



Slika 4.8. *Pregled obavijesti na podstarnici Studenti*

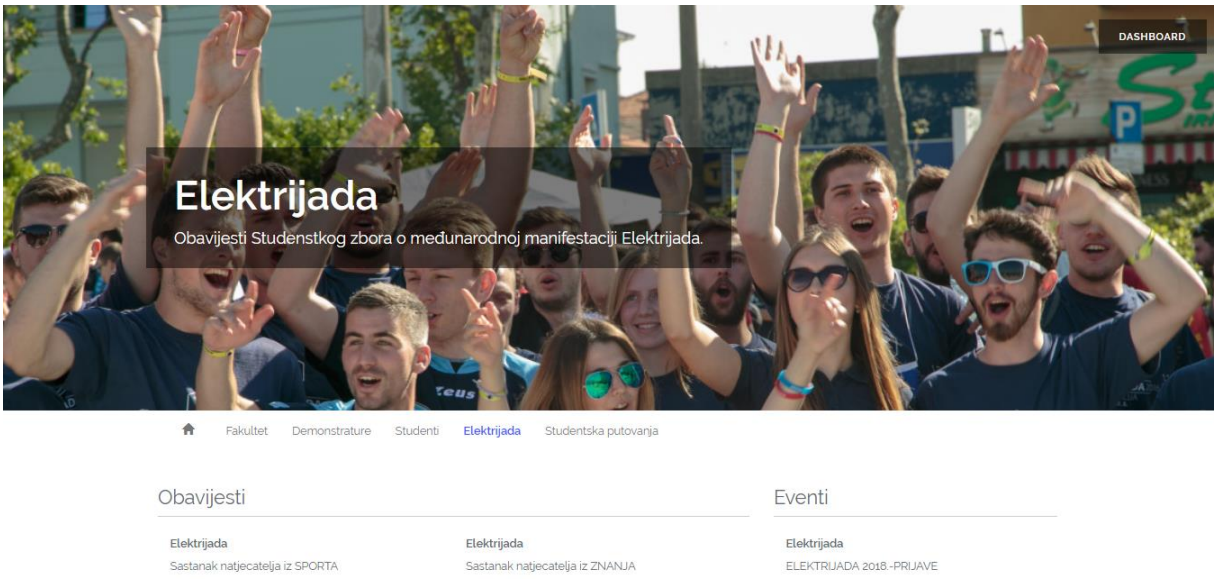
4.1.5. Elektrijada

Elektrijada je poznata međunarodna manifestacija studenata elektrotehničkih fakulteta, s područja Balkana, na kojoj se već pedesetak godina studenti uspješno natječu u znanju i sportu. Elektrijada je najveći godišnji skup studenata elektrotehnike u Europi, a okuplja oko 2000 studenata.

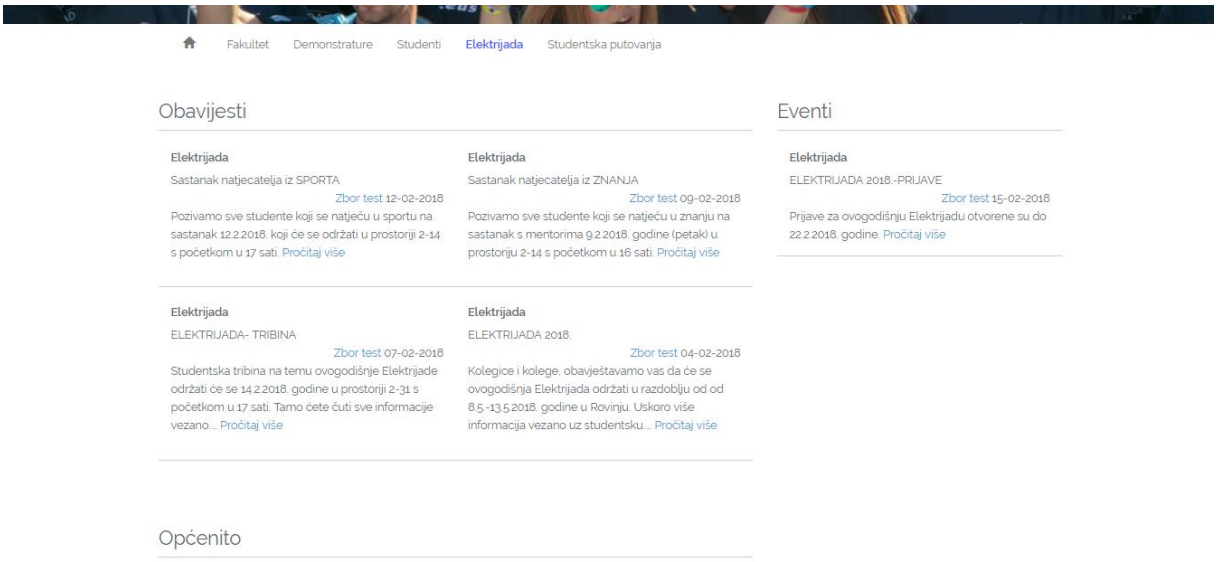
Elektrijada je natjecanje u kojem sudionici, natječući se u svojim disciplinama, razvijaju timski duh, međusobnu povezanost i samim time osjećaj da su dio jedne veće zajednice. Budući da se samo natjecanje odvija 5 dana na jedinstvenoj lokaciji gdje su svi sudionici smješteni i izravno povezani, stvara se prilika za druženje, razmjenu priča i iskustava koja svima ostaju u lijepom sjećanju. Elektrijada kao takva predstavlja dio studiranja koji svaki student zaslužuje i može osjetiti ako pokaže malo odvažnosti.

FERIT je do sada na Elektrijadi sudjelovao 4 puta, a tradiciju planira nastaviti. Elektrijada je među studentima FERIT-a postala neizostavna tema, ali i događaj kojemu se svi studenti vesele. Studentski zbor glavni je organizator Elektrijade, pa je ova podstanica važna karika portala.

Na ovoj podstanici studenti se mogu upoznati sa definicijom Elektrijade, ali i pratiti obavijesti, kao i popunjavati prijave za ovu manifestaciju nakon što ih Studentski zbor otvori.



Slika 4.9. *Izgled podstranice Elektrijada*

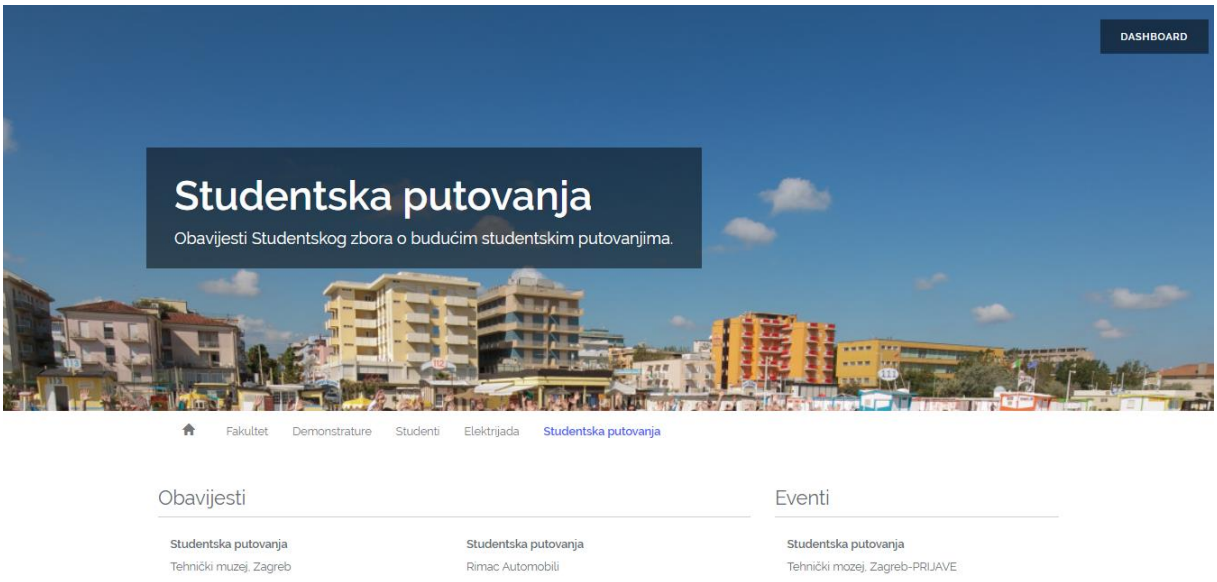


Slika 4.10. *Pregled obavijesti i otvorenih događaja na podstranici Elektrijada*

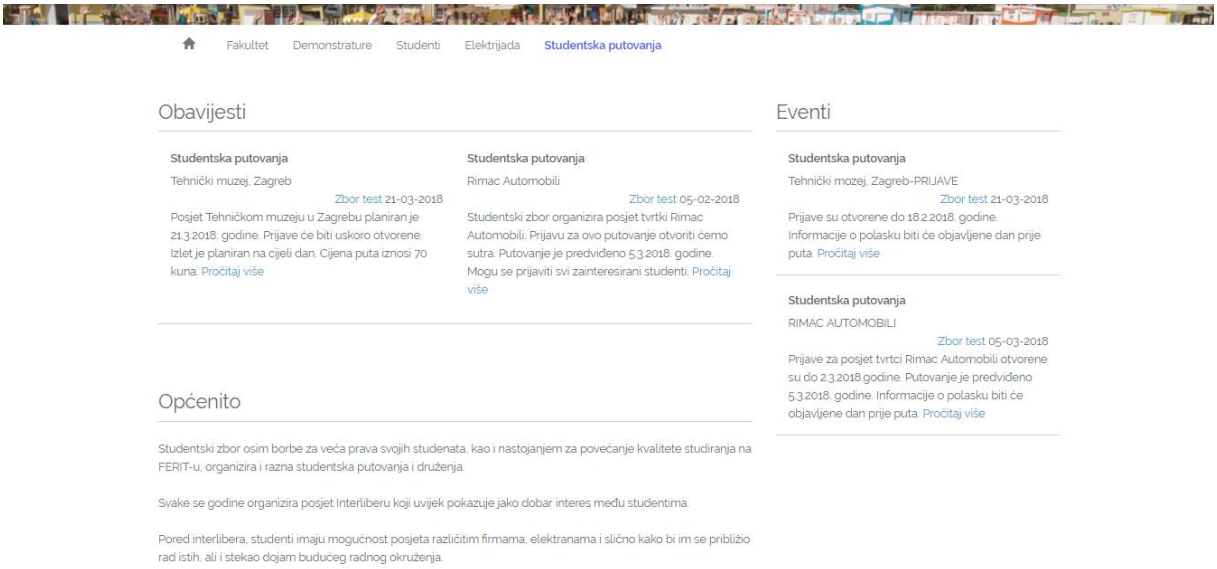
4.1.6. Studentska putovanja

Studentski zbor tijekom jedne akademske godine organizira nekoliko studentskih putovanja koja mogu biti obrazovnog ili zabavnog sadržaja. Studenti FERIT-a svake godine odlaze na Interliber, a pored toga često su za njih organizirani posjeti određenim tvrtkama, poput Rimac Automobili, Ericsson Nikola Tesla i dr., elektranama, muzejima i slično.

Ova podstranica predstavlja mjesto gdje će studenti imati uvid o studentskim putovanjima koja za njih organizira Studentski zbor, a nakon otvaranja prijava, studenti će se tamo moći i prijaviti za određeno putovanje.



Slika 4.11. *Izgled podstranice Studentska putovanja*

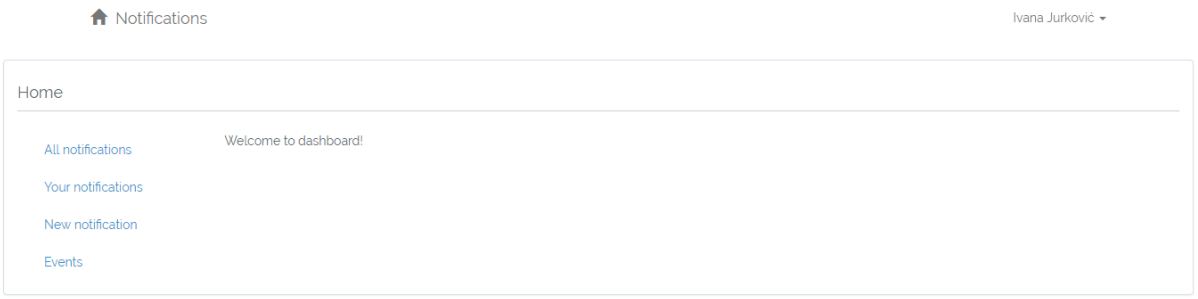


Slika 4.12. *Pregled obavijesti i događaja na podstranici Studentska putovanja*

4.2. Dashboard

Dashboard je dio web stranice koji je vidljiv nakon Login-a korisnika. Dashboard je mjesto na kojem logirani korisnik može pregledati sve oglase koje je on postavio ili sve oglase koji su općenito postavljeni na stranici od strane njega i drugih korisnika. Unutar Dashboard-a logirani korisnik ima mogućnost kreiranja novih obavijesti, tj oglasa, koje želi postaviti na portalu. Također, korisnik ima mogućnost uređivanja vlastitih obavijesti ili brisanje istih.

Pored objavljenih obavijesti, korisnik ima pregled i objavljenih događaja (*engl. Events*) na koje se može prijaviti. Izgled, kao i funkcije unutar Dashboarda ovise o kategoriji prijavljenog korisnika. Studentski zbor, kao glavni administrator stranice, ima mogućnost dodavanja nove kategorije obavijesti ukoliko za to uvidi potrebu. Studentski zbor ujedno ima najveći broj mogućnosti unutar Dashboarda u usporedbi s ostalim registriranim korisnicima.



Slika 4.13. *Izgled Dashboard-a nakon Login-a studenta*

5. FUNKCIONALNOST STRANICE STUDENTSKOG ZBORA

Stranica Studentskog zbora zamišljena je kao mjesto na kojemu će Studentski zbor postavljati obavijesti vezane za studente. Također, obavijesti će moći postavljati Prodekan za nastavu i studente, kao i studenti FERIT-a, ali samo pod određenim kategorijama. Postoje tri kategorije registriranih korisnika:

1. Studentski zbor
2. Prodekan za nastavu i studente
3. Studenti

RoleTableSeeder.php

```php
<?php

use Illuminate\Database\Seeder;
use \HttpOz\Roles\Models\Role;

class RoleTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        //Role id 1
        $prodekanRole = Role::create([
            'name' => 'Prodekan',
            'slug' => 'prodekan',
            'description' => 'Prodekan role',
            'group' => 'default'
        ]);

        //Role id 2
        $zborRole = Role::create([
            'name' => 'Studentski zbor role',
            'slug' => 'zbor',
            'description' => 'Studentski zbor role',
            'group' => 'default'
        ]);

        //Role id 3
        $studentRole = Role::create([
            'name' => 'Student',
            'slug' => 'student',
            'description' => 'Student role',
            'group' => 'default'
        ]);
    }
}
```

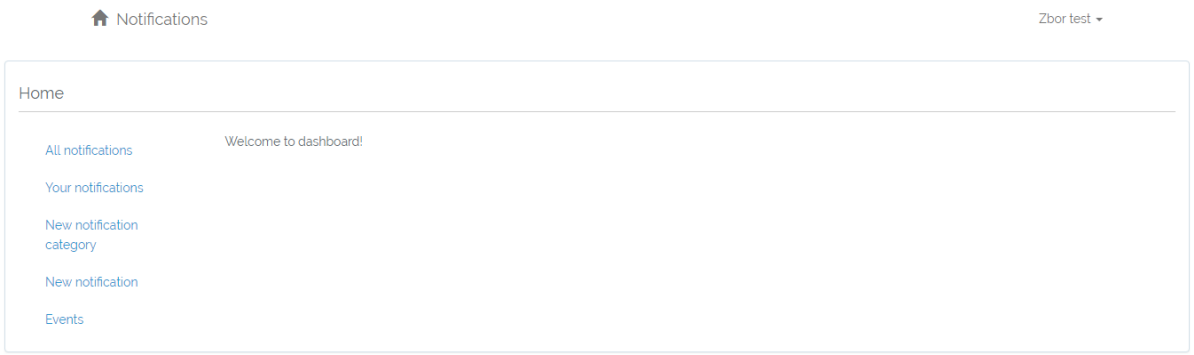
Slika 5.1. *Definiranje prava korisnika na Portalu unutar Dashboarda*

5.1. Studentski zbor kao registrirani korisnik

Studentski zbor predstavlja glavnog uređivača, tj. administratora stranice te kao takav posjeduje najviše ovlasti na stranici.

Ulaskom na Dashboard Studentski zbor posjeduje sljedeće mogućnosti:

- može pregledati sve obavijesti postavljene na stranici,
- može pregledati sortirane samo vlastite obavijesti,
- može kreirati obavijesti pod bilo kojom kategorijom,
- može uređivati sve obavijesti postavljene na stranici,
- može obrisati bilo koju obavijest postavljenu na stranici,
- može kreirati novu kategoriju obavijesti,
- može otvarati događaje *(engl. Event)* poput Elektrijade i studentskih putovanja.



Slika 5.2. *Izgled Dashboard-a Studentskog zbora*

```
<div class="row">
    <div class="col-md-12">
        <div class="panel panel-default">
            <div class="panel-body">

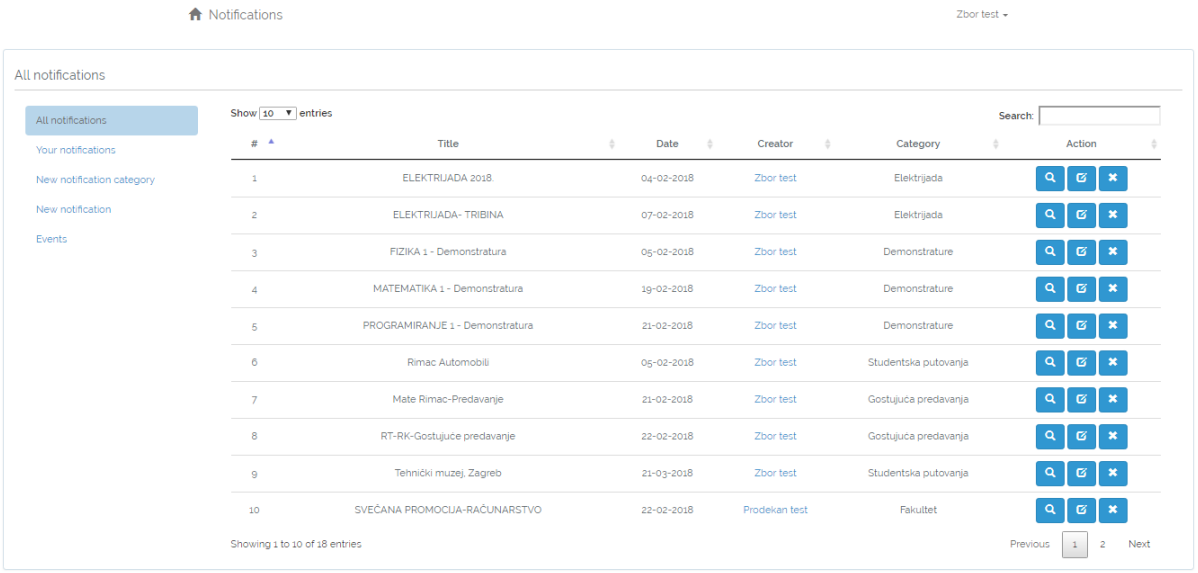
                <div class="row">
                    <h4>Home</h4>
                    <hr>
                </div>

                <div class="col-md-2">
                    <ul class="nav flex-column">
                        <li class="nav-item">
                            <a class="nav-link" href="{{ route('all.notifications') }}">All notifications</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" href="{{ route('your.notifications') }}">Your notifications</a>
                        </li>
                        @if(Auth::user()->isRole('zbor'))
                            <li class="nav-item">
                                <a class="nav-link" href="{{ route('insert.category') }}">New notification category</a>
                            </li>
                        @endif
                        <li class="nav-item">
                            <a class="nav-link" href="{{ route('home.notification') }}">New notification</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" href="{{ route('index.events') }}">Events</a>
                        </li>
                    </ul>
                </div>
                <div class="col-md-10">
                    Welcome to dashboard!
                </div>

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

Slika 5.3. *Definiranje izgleda Dashboarda nakon Logina-a*

Iz koda na *Slici 5.3.* vidimo definirane mogućnosti unutar Dashboarda nakon što se korisnik prijavio na portal. Ukoliko je prijavljeni korisnik Studentski zbor, jedino on ima mogućnost dodavanja nove kategorije *(engl. New notification category*) obavijesti na portalu. Iz tog razloga mu je u kodu dozvoljen prikaz ove mogućnosti unutar Dashboarda.



Slika 5.4. *Pregled svih obavijesti*

```
@foreach(App\Notification::all() as $c)
    <tr>
        <td>{{ $c->id }}</td>
        <td>{{ $c->title }}</td>
        <td>{{ Carbon\Carbon::parse($c->date)->format('d-m-Y') }}</td>
        <td>
            <a href="javascript:void(0)" data-id="{{ $c->notificationCreator->id }}"
            class="user-details-link">{{ $c->notificationCreator->name }}</a>
        </td>
        <td>{{ App\Category::findOrFail($c->category_id)->title }}</td>
        {{--<td><a href="{{ url('/')}}/system/storage/app/documents/{{ $c->document }}
        " target="_blank">{{ $c->document }}</a></td>--}}
        <td>
            <button data-category="{{ App\Category::findOrFail($c->category_id)->
            title }}" data-id="{{ $c->id }}" data-title="{{ $c->title }}" data-date="
            {{ $c->date }}" data-text="{{ $c->text }}" data-doc="{{ $c->document }}"
            type="button" class="btn btn-primary notification-view">
                <span class="glyphicon glyphicon-search" aria-hidden="true"></span>
            </button>
            @if($c->notificationCreator->id == Auth::user()->id || Auth::user()->
            isRole('zbor'))
                <button data-category="{{ App\Category::findOrFail($c->category_id)->
                title }}" data-id="{{ $c->id }}" data-title="{{ $c->title }}"
                data-date="{{ $c->date }}" data-text="{{ $c->text }}" data-doc="{{ $c-
                >document }}" type="button" class="btn btn-primary notification-edit">
                    <span class="glyphicon glyphicon-edit" aria-hidden="true"></span>
                </button>
                <button data-category="{{ App\Category::findOrFail($c->category_id)->
                title }}" data-id="{{ $c->id }}" data-title="{{ $c->title }}"
                data-date="{{ $c->date }}" data-text="{{ $c->text }}" data-doc="{{ $c-
                >document }}" type="button" class="btn btn-primary notification-remove
                ">
                    <span class="glyphicon glyphicon-remove" aria-hidden="true"></span
                    >
                </button>
            @endif
        </td>
    </tr>
@endforeach
```

Slika 5.5. *Ispis i definiranje mogućnosti pojedinih obavijesti*

Studentski zbor jedini je korisnik stranice koji ima mogućnost uređivanja i brisanja bilo koje obavijesti na portalu, bila ona objavljenja od strane Zbora ili od nekog drugog korisnika. Na *Slici 5.5.* definiran je prikaz *buttona* za te mogućnosti dok su na *Slikama 5.6. i 5.7***.** prikazani kontroleri za te mogućnosti.

```
public function updateNotification(Request $request){

    $notification = Notification::findOrFail($request->nid);

    if($request->file('document_update') != ''){

        if($notification->document != null){
            $filename_to_delete = $notification->document;
            unlink(storage_path('app/documents/'.$filename_to_delete));
        }

        $file = $request->file('document_update');
        $extension = $file->getClientOriginalExtension();
        $timestamp = Carbon::now()->timestamp;
        $filename = $timestamp.'.'.$extension;
        $path = $file->storeAs('documents', $filename);
    }else{
        $filename = null;
    }

    $date = explode('-', $request->date_update);

    $notification->update([
        'title'      =>  $request->title_update,
        'text'       =>  $request->text_update,
        'date'       =>  $date[2].'-'.$date[1].'-'.$date[0],
        'document'   =>  $filename
    ]);

    $category = Category::findOrFail($request->category_update);

    $notification->notificationCategory()->associate($category);
    $notification->save();

    return redirect()->back()->with('msg', 'Successfully updated notification '.$
        notification->title);
}
```

Slika 5.6. *Prikaz kontrolera za ažuriranje (engl Update) obavijesti*

```
public function removeNotification(Request $request){
    $notification = Notification::findOrFail($request->nid);
    $msg_title = $notification->title;

    if($notification->document != null){
        $filename = $notification->document;
        unlink(storage_path('app/documents/'.$filename));
    }

    $notification->delete();

    return redirect()->back()->with('msg_delete', 'Successfully deleted notification '.$
        msg_title);
}
```

Slika 5.7. *Prikaz kontrolera za brisanje obavijesti*

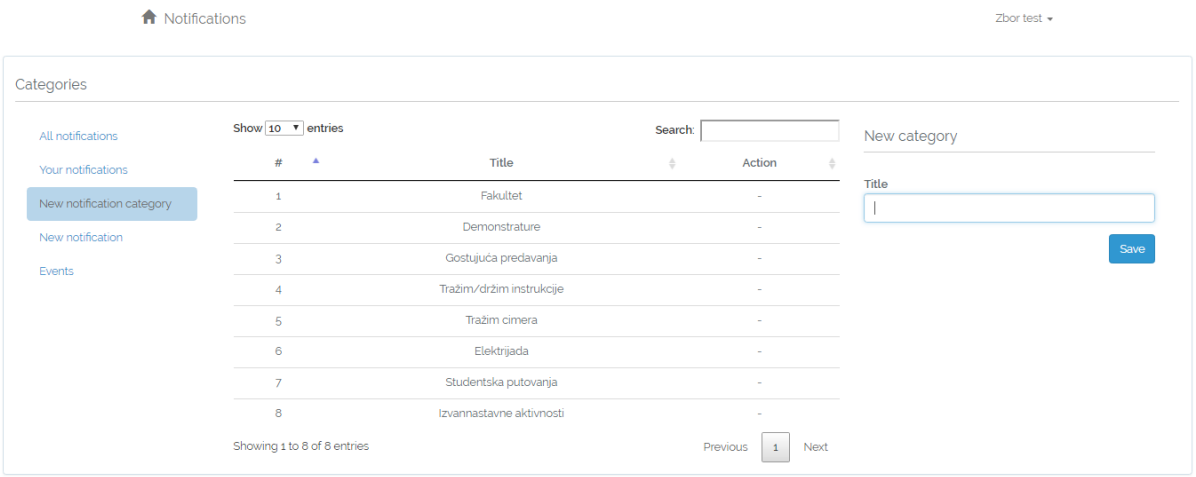
Postoji osam glavnih kategorija obavijesti na Portalu za Studentski zbor i studente FERT-a. Te kategorije, ovisno o vijesti koje donose, raspoređene su na postojeće podstranice. Svaka podstranica prikazuje vijesti iz određene kategorije.

```php
use Illuminate\Database\Seeder;

class NotificationCategorySeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        App\Category::create([
            'title' => 'Fakultet',  //zbor, prodekan
        ]);
        App\Category::create([
            'title' => 'Demonstrature', //zbor
        ]);
        App\Category::create([
            'title' => 'Gostujuća predavanja',  //zbor
        ]);
        App\Category::create([
            'title' => 'Tražim/držim instrukcije',  //zbor, studenti
        ]);
        App\Category::create([
            'title' => 'Tražim cimera', //zbor, studenti
        ]);
        App\Category::create([
            'title' => 'Elektrijada',   //zbor
        ]);
        App\Category::create([
            'title' => 'Studentska putovanja',  //zbor
        ]);
        App\Category::create([
            'title' => 'Izvannastavne aktivnosti',  //zbor
        ]);
    }
}
```

Slika 5.8. *Definiranje zadanih kategorija obavijesti*

Studentski zbor ima mogućnost dodavanja nove kategorije obavijesti. Obavijesti iz naknadno dodane kategorije prikazivati će se na Naslovnoj stranici i unutar Dashboarda pod pregledom svih obavijesti.



Slika 5.9. *Prikaz i dodavanje nove kategorije obavijesti*

```
<div class="col-md-4">
    <div class="row">
        <h4>New category</h4>
        <hr>
    </div>
    <div class="row">
        <form class="form-horizontal" method="POST" action="{{ route('insert.category') }}">
            {{ csrf_field() }}

            <div class="form-group{{ $errors->has('name') ? ' has-error' : '' }}">
                <div class="row">
                    <label for="category-title" class="control-label">Title</label>
                </div>

                <div class="row">
                    <div class="col-md-12 padding-0">
                        <input id="category-title" type="text" class="form-control" name="
                        category_title" value="" required autofocus>

                        @if ($errors->has('name'))
                            <span class="help-block">
                                <strong>{{ $errors->first('name') }}</strong>
                            </span>
                        @endif
                    </div>
                </div>
            </div>

            <div class="form-group">
                <div class="col-md-12 padding-0 text-right">
                    <button type="submit" class="btn btn-primary">
                        Save
                    </button>
                </div>
            </div>
        </form>
    </div>
</div>
```

Slika 5.10. *Definiranje izgleda forme za dodavanje nove kategorije obavijesti*

```
public function storeCategory(Request $request){
    if(!Auth::user()->isRole('zbor')){
        return redirect()->back();
    }

    $category = Category::create([
        'title' => $request->category_title
    ]);

    return redirect()->back()->with('msg', 'Successfully added category '.$category->title);
}

public function updateCategory(Request $request){
    if(!Auth::user()->isRole('zbor')){
        return redirect()->back();
    }

    $category = Category::findOrFail($request->cid);

    $category->update([
        'title' => $request->category_title_update
    ]);

    return redirect()->back()->with('msg', 'Successfully updated category '.$category->title);
}

public function removeCategory(Request $request){
    if(!Auth::user()->isRole('zbor')){
        return redirect()->back();
    }

    $category = Category::findOrFail($request->cid);
    $msg_title = $category->title;
    $category->delete();

    return redirect()->back()->with('msg_delete', 'Successfully deleted category '.$msg_title);
```

Slika 5.11. *Prikaz kontrolera za dodavanje, uređivanje i brisanje kategorije koja je naknadno dodana*

Studentski zbor, kao i svi ostali korisnici, imaju određene kategorije pod koje mogu dodavati nove obavijesti na Portalu koje su definirane po uvjetima na sljedećoj slici.

```
<div class="form-group{{ $errors->has('category') ? ' has-error' : '' }}">
    <div class="row">
        <label for="category" class="control-label">Category</label>
    </div>

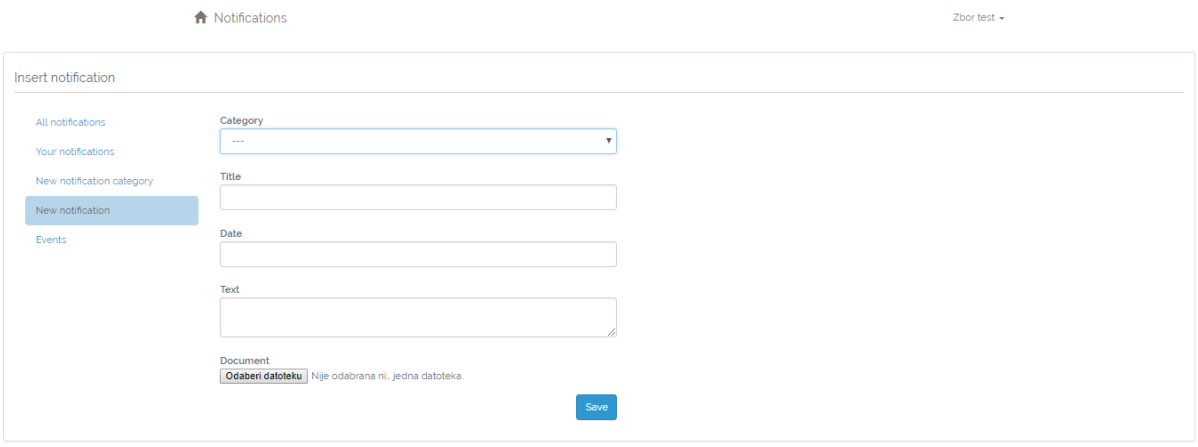
    <div class="row">
        <div class="col-md-5 padding-0">
            <select id="category" class="form-control" name="category" required autofocus>
                <option value="">---</option>
                @if(Auth::user()->isRole('zbor'))
                    @foreach(App\Category::all() as $c)
                        <option value="{{ $c->id }}">{{ $c->title }}</option>
                    @endforeach
                @endif

                @if(Auth::user()->isRole('prodekan'))
                    <option value="{{ App\Category::findOrFail(1)->id }}">{{
                    App\Category::findOrFail(1)->title }}</option>
                @endif

                @if(Auth::user()->isRole('student'))
                    @foreach(App\Category::where('id', 4)->orWhere('id', 5)->get() as $c)
                        <option value="{{ $c->id }}">{{ $c->title }}</option>
                    @endforeach
                @endif
            </select>

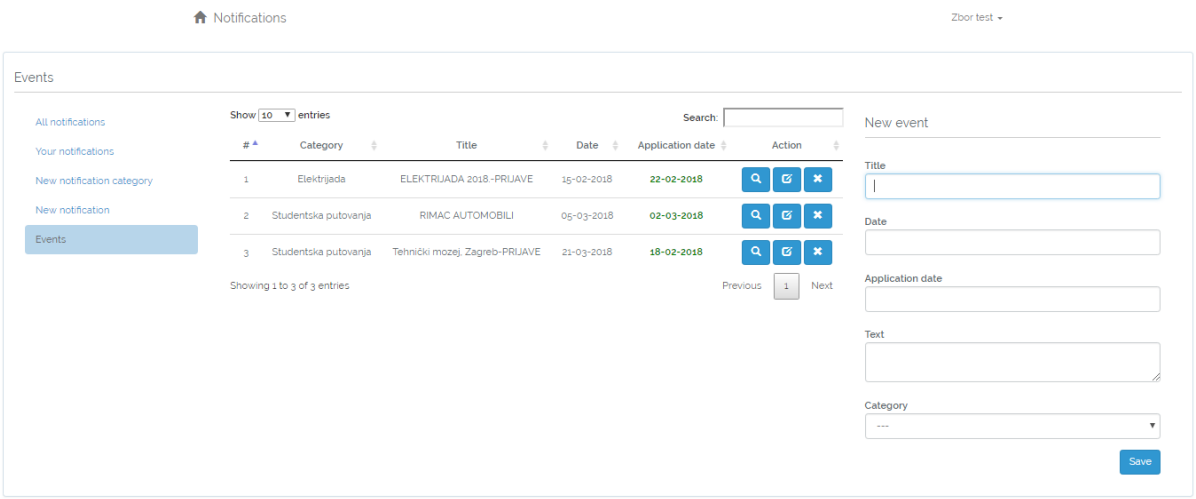
            @if ($errors->has('category'))
                <span class="help-block">
                    <strong>{{ $errors->first('category') }}</strong>
                </span>
            @endif
        </div>
    </div>
</div>
```

Slika 5.12. *Definiranje kategorija obavijesti za svakog korisnika portala*



Slika 5.13. *Izgled forme za dodavanje nove obavijesti*

Studentski zbor jedini je korisnik portala koji ima mogućnost dodavanja novog događaja *(Engl. Event)*. Događaj se može kreirati pod dvije kategorije*: Elektrijada* i *Studentska putovanja.*



Slika 5.14. *Prikaz i dodavanje novog događaja (engl. Event-a)*

```php
<?php

use Illuminate\Database\Seeder;

class EventCategorySeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        App\EventCategory::create([
            'title' => 'Elektrijada'
        ]);

        App\EventCategory::create([
            'title' => 'Studentska putovanja'
        ]);
    }
}
```

Slika 5.15. *Definiranje dviju zadanih kategorija događaja*

```
@if(Auth::user()->isRole('zbor'))
    <div class="col-md-4">
        <div class="row">
            <h4>New event</h4>
            <hr>
        </div>
        <div class="row">
            <form class="form-horizontal" method="POST" action="{{ route('insert.event') }}">
                {{ csrf_field() }}

                <div class="form-group{{ $errors->has('event_title') ? ' has-error' : '' }}">
                    <div class="row">
                        <label for="event-title" class="control-label">Title</label>
                    </div>

                    <div class="row">
                        <div class="col-md-12 padding-0">
                            <input id="event-title" type="text" class="form-control" name="event_title
                            " value="" required autofocus>

                            @if ($errors->has('event_title'))
                                <span class="help-block">
                                    <strong>{{ $errors->first('event_title') }}</strong>
                                </span>
                            @endif
                        </div>
                    </div>
                </div>

                <div class="form-group{{ $errors->has('event_date') ? ' has-error' : '' }}">
                    <div class="row">
                        <label for="event-date" class="control-label">Date</label>
                    </div>

                    <div class="row">
                        <div class="col-md-12 padding-0">
                            <input id="event-date" type="text" class="form-control datepicker" name="
                            event_date" value="" required autofocus>

                            @if ($errors->has('event_date'))
                                <span class="help-block">
                                    <strong>{{ $errors->first('event_date') }}</strong>
                                </span>
                            @endif
                        </div>
                    </div>
                </div>

                <div class="form-group{{ $errors->has('event_application_date') ? ' has-error' : '' }}
                ">
                    <div class="row">
                        <label for="event-application-date" class="control-label">Application date</
                        label>
                    </div>

                    <div class="row">
                        <div class="col-md-12 padding-0">
                            <input id="event-application-date" type="text" class="form-control
                            datepicker" name="event_application_date" value="" required autofocus>

                            @if ($errors->has('event_application_date'))
                                <span class="help-block">
                                    <strong>{{ $errors->first('event_application_date') }}</strong>
                                </span>
                            @endif
                        </div>
                    </div>
                </div>

                <div class="form-group{{ $errors->has('event_text') ? ' has-error' : '' }}">
                    <div class="row">
                        <label for="event-text" class="control-label">Text</label>
                    </div>

                    <div class="row">
                        <div class="col-md-12 padding-0">
                            <textarea id="event-text" class="form-control" name="event_text" required
                            autofocus></textarea>

                            @if ($errors->has('event_text'))
                                <span class="help-block">
                                    <strong>{{ $errors->first('event_text') }}</strong>
                                </span>
                            @endif
                        </div>
                    </div>

                    <div class="form-group{{ $errors->has('event_category') ? ' has-error' : '' }}">
                        <div class="row">
                            <label for="event-category" class="control-label">Category</label>
                        </div>

                        <div class="row">
                            <div class="col-md-12 padding-0">
                                <select id="event-category" class="form-control" name="event_category"
                                required autofocus>
                                    <option value="">---</option>
                                    @foreach(App\EventCategory::all() as $ec)
                                        <option value="{{ $ec->id }}">{{ $ec->title }}</option>
                                    @endforeach
                                </select>

                                @if ($errors->has('event_category'))
                                    <span class="help-block">
                                        <strong>{{ $errors->first('event_category') }}</strong>
                                    </span>
                                @endif
                            </div>
                        </div>
                    </div>

                    <div class="form-group">
                        <div class="col-md-12 padding-0 text-right">
                            <button type="submit" class="btn btn-primary">
                                Save
                            </button>
                        </div>
                    </div>
                </form>
            </div>
        </div>
```

Slika 5.16. *Definiranje forme za dodavanje novog događaja*

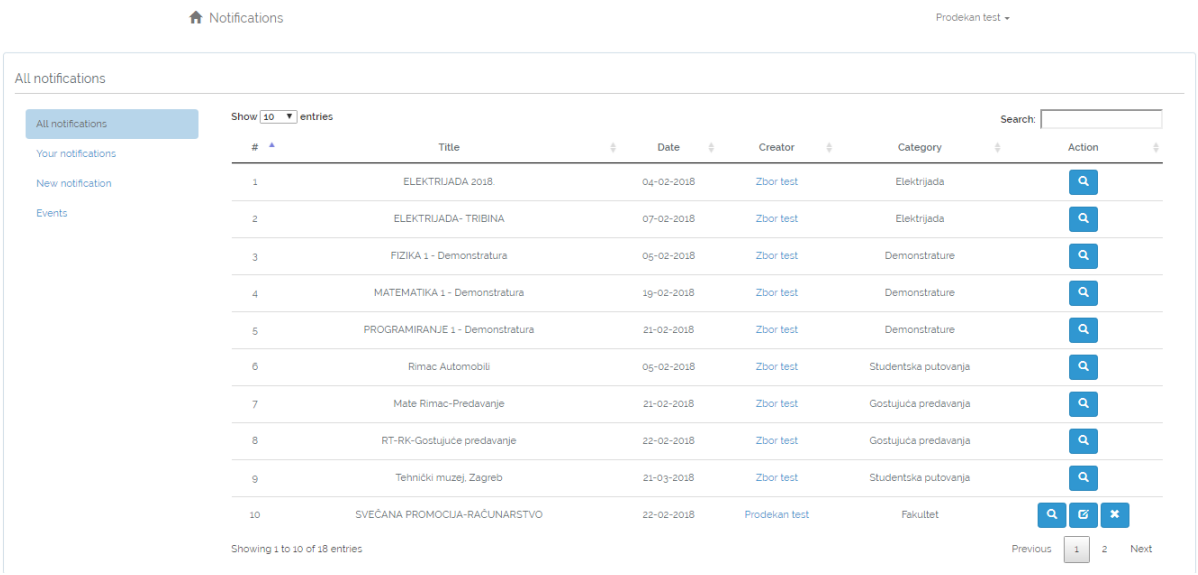
5.2. Prodekan za nastavu i studente kao registrirani korisnik

Uloga Prodekana za nastavu i studente je dodavanje obavijesti na stranici pod kategorijom Fakultet. Budući da prodekan iz prve ruke ima uvid u sva događanja na fakultetu, on će te obavijesti i moći postavljati na stranici. Prodekan nema mogućnost dodavanja sadržaja pod drugim kategorijama, budući da su one usko vezane uz zadaće Studentskog zbora.

Ulaskom na Dashboard Prodekan ima slijedeće mogućnosti:

- može pregledati sve obavijesti postavljene na stranici,
- može pregledati sortirane samo vlastite obavijesti,
- može kreirati obavijesti pod kategorijom Fakultet,
- može uređivati samo vlastite obavijesti,
- može obrisati samo vlastite obavijesti.

Kao što je prethodno pokazano u kodu, samo Studentski zbor može uređivati i brisati sve obavijesti. Kada je kao korisnik na Portal prijavljen Prodekan, on može uređivati i brisati samo obavijesti koje je on postavio.

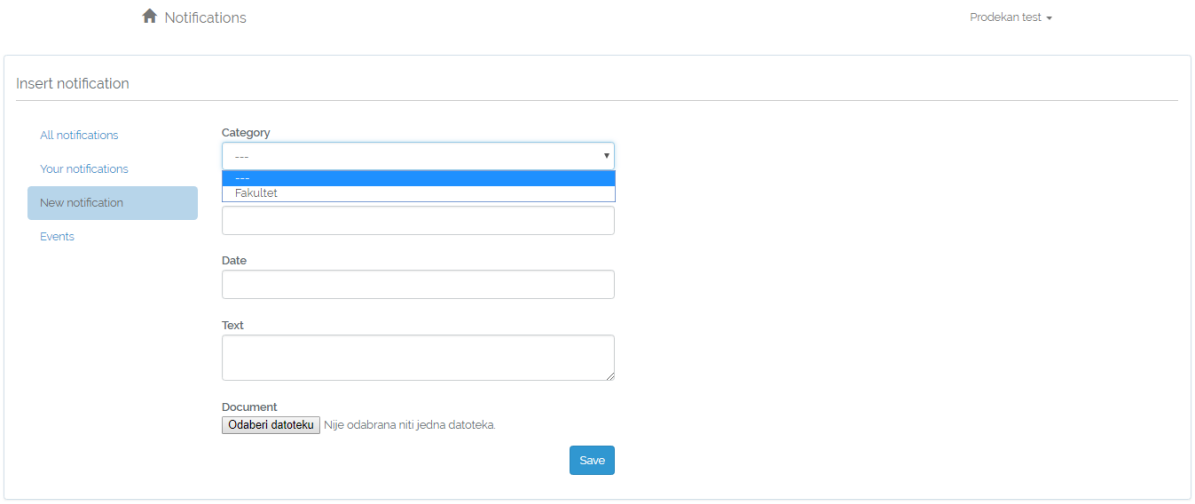


Slika 5.17. *Pregled svih obavijesti*

Prodekan može postavljati nove obavijesti samo pod kategorijom Fakultet koje će se prikazivati na podstranici Fakultet.

```
@if(Auth::user()->isRole('prodekan'))
    <option value="{{ App\Category::findOrFail(1)->id }}">{{
    App\Category::findOrFail(1)->title }}</option>
@endif
```

Slika 5.18. *Definiranje kategorije pod kojom može objavljivati Prodekan*



Slika 5.19. *Prikaz kreiranja nove obavijesti od strane Prodekana*

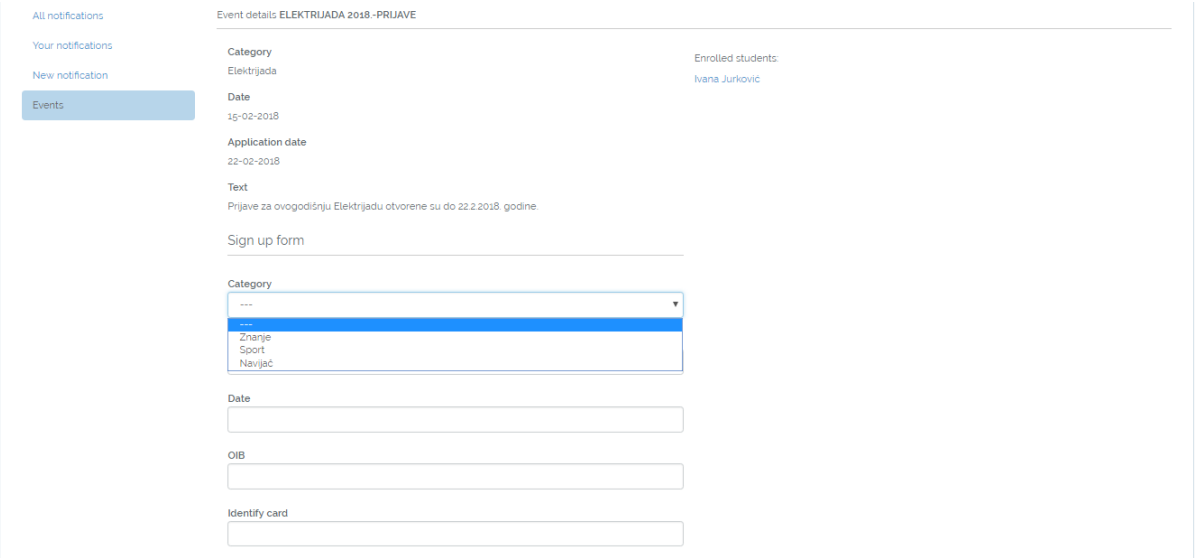
5.3. Student kao registrirani korisnik

Portal Studentskog zbora za studente FERIT-a služi kao mjesto gdje će studenti pravovremeno dobivati sve informacije koje se njih tiču. Pa će tako studenti po kategorijama dobivati obavjesti o događanjima na fakultetu, o održavanju demonstratura, o organizaciji i prijavama na Elektrijadu, kao i o budućim studentskim putovanjima.

Prijavljeni studenti na stranici će moći objavljivati obavijesti pod dvije kategorije: Tražim/držim instrukcije i Tražim cimera. Ove dvije kategorije pokazale su se kao izuzetno važne našim studentima jer smo svi bili u situaciji da nam je bila potrebna pomoć vezana uz gradivo, ili da smo bili u potrazi za cimerima za idući akademsku godinu. Iz tog razloga studenti imaju mogućnost objavljivanja u tim kategorijama.

Ulaskom na Dashboard studenti imaju sljedeće mogućnosti:

- može pregledati sve obavijesti postavljene na stranici,
- može pregledati sortirane samo vlastite obavijesti,
- može kreirati obavijesti pod kategorijama Tražim/držim instrukcije i Tražim cimera,
- može uređivati samo vlastite obavijesti,
- može obrisati samo vlastite obavijesti,
- može se prijaviti na događaje (*engl. Event*) koje je otvorio Studentski zbor



Slika 5.20. *Izgled obrasca za prijavu studenta na događaj Elektrijada*

```
<h4>Sign up form</h4>
<hr>

<div class="form-group{{ $errors->has('event_form_category') ? ' has-error' : '' }}">
    <div class="row">
        <label for="event-form-category" class="control-label">Category</label>
    </div>

    <div class="row">
        <div class="col-md-12 padding-0">
            <select id="event-form-category" class="form-control" name="event_form_category" required autofocus>
                <option value="">---</option>
                <option value="1">Znanje</option>
                <option value="2">Sport</option>
                <option value="3">Navijač</option>
            </select>

            @if ($errors->has('event_form_category'))
                <span class="help-block">
                    <strong>{{ $errors->first('event_form_category') }}</strong>
                </span>
            @endif
        </div>
    </div>
</div>

<div class="form-group{{ $errors->has('event_form_name') ? ' has-error' : '' }}">
    <div class="row">
        <label for="event-form-name" class="control-label">Name</label>
    </div>

    <div class="row">
        <div class="col-md-12 padding-0">
            <input id="event-form-name" type="text" class="form-control" name="event_form_name" value="" required
            autofocus>

            @if ($errors->has('event_form_name'))
                <span class="help-block">
                    <strong>{{ $errors->first('event_form_name') }}</strong>
                </span>
            @endif
```

Slika 5.21. *Dio koda obrasca za prijavu na događaj Elektrijada*

```php
public function regElekEvent(Request $request){
    $event = Event::findOrFail($request->eid);
    $user = Auth::user();

    // If reg == 0 => sing out else 1 => sing in
    if($request->reg == 0){

        $elek_form_obj = EventForm::findOrFail($user->userElekEvent->where('pivot.event_id', $event->id)->first()->
            pivot->elek_form_id);

        $user->userElekEvent()->detach($event);

        $elek_form_obj->delete();

        return redirect()->back()->with('msg', 'You have successfully signed out event '.$event->title);
    }else{

        $date = explode('-', $request->event_form_date);

        $apply_request = '';
        switch ($request->event_form_category) {
            case 1:
                foreach($request->event_form_apply as $key => $value){
                    $apply_request .= $value.'//';
                }
                $apply_request = substr($apply_request, 0, -2);
                break;
            case 2:
                $apply_request = $request->event_form_apply;
                break;
            case 3:
                $apply_request = null;
                break;
            default:
                $apply_request = null;
        }

        $event_form = EventForm::create([
            'category'          => $request->event_form_category,
            'name'              => $request->event_form_name,
            'date'              => $date[2].'-'.$date[1].'-'.$date[0],
            'oib'               => $request->event_form_oib,
            'identify_card'     => $request->event_form_identify_card,
            'address'           => $request->event_form_address,
            'year'              => $request->event_form_year,
            'contact'           => $request->event_form_contact,
            'email'             => $user->email,
            'size_shirt'        => $request->event_form_size_shirt,
            'apply'             => $apply_request,
        ]);

        $user->userElekEvent()->attach($event, ['elek_form_id' => $event_form->id]);

        return redirect()->back()->with('msg', 'You have successfully signed up event '.$event->title);
    }
}
```

Slika 5.22. *Prikaz kontrolera za registraciju na događaj Elektrijada*

6. ZAKLJUČAK

Portal za Studentski zbor i studente FERIT-a mjesto je gdje studenti imaju uvid u sve informacije koje za njih objavljuje Studentski zbor, ali i Prodekan Fakulteta.

Studentski zbor do sada je imao samo Facebook stranicu putem koje su komunicirali sa studentima i gdje su otvarali natječaje. Glavni nedostatak takve komunikacije bio je u tome što obavijesti nisu bile sortirane po kategorijama, niti su bile pregledne. Također, nemaju svi studenti Facebook pa sam kao članica Studentskog zbora došla na ideju izrade upravo ovakve web stranice.

Glava prednost ovog Portala je sortiranost obavijesti po kategorijama i kao takva olakšava potragu studentima za nekom od prethodno objavljene obavijesti. Putem ovog portala studenti se mogu prijavljivati na događaje, poput Elektrijade i drugih studentskih putovanja, koje za njih organizira Studentski zbor i Fakultet. Studentski zbor ima uvid u prijave, kao i podatke svakog prijavljenog studenta. Također, studenti mogu objavljivati obavijesti pod kategorijama Tražim/držim instrukcije i Tražim cimera.

Posjedovanjem navedenih karakteristika, ovaj portal će se, bez problema, pokazati kao bolje sredstvo za komunikaciju i oglašavanje studenata od do sada postojeće Facebook stranice Studentskog zbora.

LITERATURA

[1] World Wide Web, https://webfoundation.org/about/vision/history-of-the-web/, (4.2.2018.)

[2] Web-stranica, https://www.computerhope.com/jargon/w/webpage.htm, (4.2.2018.)

[3] HTML, http://www.oblakznanja.com/2013/04/html-dokument-i-osnove-html-jezika/, (5.2.2018.)

[4] CSS, https://bib.irb.hr/datoteka/532594.Skripta_-_Uvod_u_xhtml_html_i_css.pdf, (5.2.2018.)

[5] PHP, http://www.netakademija.hr/sto-je-php/, (5.2.2018.)

[6] jQuery, https://www.popwebdesign.net/popart_blog/2010/06/jquery-sta-je-i-cemu-sluzi/, (5.2.2018.)

[7] JavaScript, http://www.tutorialspoint.com/javascript/javascript_overview.htm, (6.2.2018.)

[8] Bootstrap, https://getbootstrap.com/docs/4.0/getting-started/introduction/, (6.2.2018.)

[9] Laravel, https://laravel.com/docs/5.6, (6.2.2018.)

[10] MySQL, http://razno.sveznadar.info/10-doc-PDF/2_MySQL.pdf, (6.2.2018.)

[11] Laravel tutorial, How to build a blog with Laravel, https://www.youtube.com/watch?v=R8B4og-BeCk&list=PLwAKR305CRO-Q90J---jXVzbOd4CDRbVx, (5.1.2018.)

[12] Ducket, J., HTML&CSS, Design and build websites, (8.11.2011.)

[13] Lerdof, R., Tatroe, K., Maclntyre, R., Programming PHP, Creating dinamic web pages, (7.2.2013.)

SAŽETAK

U ovom diplomskom radu izrađena je web stranica za Studentski zbor i studente FERIT-a služeći se Laravel razvojnim okruženjem. Stranica se sastoji od 6 podstranica: Naslovna stranica, Fakultet, Demonstrature, Studenti, Elektrijada i Studentska putovanja. Postoje tri tipa registriranih korisnika: Studentski zbor, Prodekan i studenti. Studentski zbor može objavljivati obavijesti pod bilo kojom kategorijom. Prodekan može postavljati obavijesti samo pod kategorijom Fakultet, dok studenti mogu postavljati obavijesti na podstranici Studenti pod kategorijama Tražim/držim instrukcije i Tražim cimera. Studentski zbor, pored obavijesti, može otvarati i događaje na koje se studenti, nakon registracije na stranici, mogu prijaviti.

Ključne riječi: Događaji, Laravel, oglasi, registrirani korisnici, Web stranica

ABSTRACT

Title: A Website for Student Council and FERIT students

In this graduate thesis, using the Laravel Development Environment, I created a website for the Student Council and FERIT students. The page consists of 6 sub-pages: Homepage, Faculty, Demonstration, Students, manifestation Elektrijada and Student travel. There are three types of registered users: Student Council, Vice Dean and Students. The Student Council can publish notifications under any category. The Vice Dean can post notifications only under the Faculty category, while students may post notifications on the sub-page Students under the Search / Hold Instructions and Search for roommates categories. In addition to the announcement post, the Student Council can also open the events that students can apply after registering on the site.

Keywords: Advertisement, events, Laravel, registered users, Website

ŽIVOTOPIS

MATEA TOMINOVIĆ

Rođena je 11.07.1993. u Slavonskom Brodu, gdje je završila osnovu školu „Antun Mihanović“ s odličnim uspjehom. 2008. godine u Slavonskom Brodu upisuje opću gimnaziju „Matija Mesić“ koju završava 2012. godine, također s odličnim uspjehom.

2012. godine upisuje preddiplomski sveučilišni studij elektrotehnike na Elektrotehničkom fakultetu u Osijeku. Nakon završetka prve godine opredjeljuje se za smjer „Komunikacije i informatika“.

2015. godine upisuje diplomski sveučilišni studij elektrotehnike na Elektrotehničkom fakultetu u Osijeku, i odabire smjer „Mrežne tehnologije“.

2015. godine postaje potpredsjednica Studentskog zbora i članica Fakultetskog vijeća te se trudi kroz različite projekte i aktivnosti povezati studente i učiniti im studiranje zanimljivijim.

Slobodno vrijeme rado provodi u sportskom duhu, a isto tako predstavlja Fakultet na raznim sportskim natjecanjima.

Tečno govori engleski jezik i informatički je pismena.

U Osijeku, 8. veljače 2018.

Matea Tominović