Univerzitet u Beogradu

Fakultet organizacionih nauka

Laboratorija za elektronsko poslovanje





Web Igrice

Seminarski rad iz predmeta Internet tehnologije

Student: Đorđe Ražnatović 1070/18

Beograd, 2020.

Sadržaj

[1. Korisnički zahtev 3](#_Toc506685365)

[2. Opis sistema 4](#_Toc506685366)

[2.1.Opis slučajeva korišćenja 4](#_Toc506685367)

[2.1.1 Dodaj radnika 5](#_Toc506685368)

[2.1.2 Dodaj novog člana 8](#_Toc506685369)

[2.1.3 Iznajmi knjigu 10](#_Toc506685370)

[3.1. Opis arhitekture aplikacije 12](#_Toc506685371)

[3.1.1 Dijagram klasa 12](#_Toc506685372)

[3.1.2 Struktura baze 12](#_Toc506685373)

[4.1. Specifikacija REST API-ja 13](#_Toc506685374)

[3.Opis tehnologija korišćenih u aplikaciji 15](#_Toc506685375)

[3.1.HTML 15](#_Toc506685376)

[3.2.CSS 16](#_Toc506685377)

[3.3.PHP 16](#_Toc506685378)

[3.4.AJAX(Asynchronous JavaScript And XML) 17](#_Toc506685379)

[3.5.JSON 18](#_Toc506685380)

[3.6.JavaScript 19](#_Toc506685381)

[3.7.JQuery 20](#_Toc506685382)

[4.Korisničko uputstvo - Opis karakterističnih slučajeva korišćenja preko screenshot-ova aplikacije 21](#_Toc506685383)

[4.1.Početna strana(Slučaj kada niko nije ulogovan) 21](#_Toc506685384)

[4.2. Slučaj kada je ulogovan Glavni bibliotekar 22](#_Toc506685385)

[4.3. Slučaj kada je na sajtu prijavljen paor 24](#_Toc506685386)

[5.Prikaz reprezentativnih delova koda 26](#_Toc506685387)

# Korisnički zahtev

Radi se o web aplikaciji koja pruža informacije o video igricama i daje pristup sadržaju i ocenama igrica. Dakle reč je o platformi gde korisnici mogu da unose najbitnije informacije o igricama u bazu podataka i na taj način da omoguće drugima da saznaju za njihove omiljene uspomene i bezbrojne sate provedene uz ekran. Cilj je da se napravi zajednica ljudi kojih dele istu strast prema gaming-u. Postoje dve osnovne vrste uloga u web aplikaciji, a to su administrator i običan korisnik. Naša aplikacija takođe sarađuje sa prijateljskim javnim API-jem gde se takođe nalazi ogroman broj podataka o igricama i na našem web-u možete videti o čemu se radi. Preuzet je njihov sadržaj za nekoliko igrica.

# Opis sistema

## 2.1.Opis slučajeva korišćenja

## 2.1.1. Uklanjanje igrice iz baze

Karakteristični slučaji koriščenja:

1. Prijava (login) korisnika
2. Odlazak na stranicu saznaj
3. Odabir igrice iz baze
4. Čitanje prikazanih informacija
5. Uklanjanje te igrice ukoliko je došlo do izmena
6. Odjava (logout)

**Slučaj korišćenja – Uklanjanje igrice**

***NazivSK:*** Uklanjanje igrice

***Nosilac SK:*** Administrator ili korisnik

***Učesnici SK:*** Korisnik (ili administrator) i sistem

***Preduslov:*** Sistem je pokrenut.

***Osnovni scenario:***

1. Korisnik unosi podatke za logovanje
2. Sistem ga verifikuje i pokreće mu session
3. Korisnik odlazi na stranicu saznaj.php
4. Korisnik bira iz padajućeg menija igricu
5. Unosi ime igrice koju želi da ukloni u polje pored dugmeta ukloni
6. Pritiska dugme
7. Server briše igricu iz svoje baze

***Alternativni scenario:***

1.1 Ukoliko sistem ne može da prihvati logovanje, šalje poruku korisniku da logovanje nije uspešno.

Razlog za ovo može da bude ako nije dobro unet username ili password.

Renderovanje stranice

Klik na saznaj u meniju

Početak session-a

Verifikacija

Poruka o brisanju

Odgovor o brisanju

Slanje stranice

Zahtev za stranicu

Slanje podataka

Podaci za logovanje

Potvrda

Slanje forme

Popunjavanje forme

Odlazak na stranicu

Verifikacija

Logovanje

Korisnik

Klijentska strana

Serverska strana

Query za brisanje igrice

Unos imena igrice i klik

## 2.1.2 Dodaj igricu

Karakteristični slučaji koriščenja:

1. Prijava (login) korisnika
2. Odlazak na stranicu saznaj
3. Klik na dugme dodaj
4. Popunjavanje forme
5. Slanje forme
6. Odjava



**Slučaj korišćenja – Dodavanje igrice**

***NazivSK:*** Dodavanje igrice

***Nosilac SK:*** Administrator ili korisnik

***Učesnici SK:*** Korisnik (ili administrator) i sistem

***Preduslov:*** Sistem je pokrenut.

***Osnovni scenario:***

1. Korisnik unosi podatke za logovanje
2. Sistem ga verifikuje i pokreće mu session
3. Korisnik odlazi na stranicu saznaj.php
4. Korisnik pritiska dugme Dodaj
5. Pojavljuje se form sa praznim poljima za unos
6. Korisnik popunjava podatke i pritiska submit
7. Server prihvata podatke i smešta ih u bazu

***Alternativni scenario:***

1.1 Ukoliko sistem ne može da prihvati logovanje, šalje poruku korisniku da logovanje nije uspešno.

Razlog za ovo može da bude ako nije dobro unet username ili password.

Renderovanje stranice

Klik na saznaj u meniju

Početak session-a

Verifikacija

Poruka o dodavanju

Odgovor o prijemu

Podaci o igrici

Podaci o igrici

Slanje stranice

Zahtev za stranicu

Slanje podataka

Podaci za logovanje

Potvrda

Slanje forme

Popunjavanje forme

Odlazak na stranicu

Verifikacija

Logovanje

Korisnik

Klijentska strana

Serverska strana

## 3.1.2 Struktura baze

U ovom projektu korišćena je jedna baza podataka - sem.

Baza „sem“ se sastoji iz 2 tabele: users i igrice.

Tabela users se sastoji iz atributa: „id“, „username“, „password“, „role“ i „created\_at“.

Tabela igrice se sastoji iz atributa: „id“, „igrica“, „tvorac“, „platforma“, „brigraca“ i „ocena“.

## 4.1. Specifikacija REST API-ja

|  |  |
| --- | --- |
| Opis funkcije | Pregled svih korisnika |
| HTTP metoda | GET |
| URL | ebiblioteka/api/clanovi |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute clanID[int], ime[string], prezime[string], email[string], adresa[string], brojtelefona[int]  [  {  "clanID":"1",  "ime":"Mika",  "prezime":"Mikic",  "email":"mika@gmail.com",  "adresa":"Mikina 21",  "brojTelefona":"065 4578 998"  },  {  "clanID":"2",  "ime":"Milena",  "prezime":"Dravic",  "email":"milena@gmail.com",  "adresa":"Nusiceva 12",  "brojTelefona":"060 1233322"  }  ] |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Login korisnika |
| HTTP metoda | POST |
| URL | ebiblioteka/login.php |
| URL parametri | (nema) |
| HTTP body parametri | JSON objekat sa atributima username[string], password[string].  Primer:  {  "username": "Proizvoljan username",  "password" :"Proizvoljan password",  } |
| Format HTTP body parametara | application/json |
| Izlazni parametri | Nema |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Registracija korisnika |
| HTTP metoda | GET |
| URL | ebiblioteka/api/uloge |
| URL parametri | (nema) |
| HTTP body parametri | JSON objekat sa atributima ulogaID [string], nazivUlogestring].  [  {  "ulogaID":"1",  "nazivUloge":"Glavnibibliotekar"  },  {  "ulogaID":"2",  "nazivUloge":"Smrtnik(paor)"  }  ] |
| Format HTTP body parametara | application/json |
| Izlazni parametri | Nema |
| Format izlaznih parametara | application/json |

# 3.Opis tehnologija korišćenih u aplikaciji

U našem radu koristili smo sledeće tehnologije:

1. HTML
2. CSS
3. PHP
4. AJAX
5. JSON
6. JavaScript
7. JQuery

## 3.1.HTML

HTML (engl. HyperText Markup Language, jezik za označavanje hiperteksta) je opisni jezik specijalno namenjen za opis veb stranica. Pomoću njega se jednostavno mogu odvojiti elementi kao što su naslovi, paragrafi, citati i slično. Pored toga, u HTML standard su ugrađeni elementi koji detaljnije opisuju sam dokument kao što su kratak opis dokumenta, ključne reči, podaci o autoru i slično. Ovi podaci su opštepoznati kao meta podaci i jasno su odvojeni od sadržaja dokumenta.

Kao modifikovanu i pojednostavljenu verziju SGML jezika, HTML standarizuje i održava World Wide Web Consortium (W3C). Originalnu verziju HTML-a kreirao je Tim Berners-Lee, a prva zvanična verzija je izašla juna 1993 godine.

HTML jezik je sačinjen od običnog teksta i tagova. Iako se HTML znakovi (tags) često nazivaju kodom, tehnički HTML nije kod jer računarski kod su instrukcije koje od računara traže da izvrši određenu operaciju. Svrha HTML znakova je da se se struktura dokumenta "označi" tako da bi korisnikov agent (user agent), tj. internet pretraživač mogao da prepozna strukturu dokumenta i ispravno je prikaže u prozoru internet pretraževača kojeg korisnik koristi.

Svaki HTML dokument mora početi tagom <html> i završiti se tagom </html>. HTML dokument sadrži uobičajeno dva različita dela, head i body. Head sadrži informacije o dokumentu koje se ne prikazuju na ekranu. Body sadrži sve ostalo što se prikazuje kao deo web stranice.

HTML5 je jezik koji se koristi za strukturiranje i prezentovanje sadržaja za World Wide Web i predstavlja osnovnu tehnologiju Interneta. HTML5 je peta revizija HTML standarda koja se i dalje dopunjuje i razvija. Osnovni ciljevi razvijanja ove poslednje verzije HTML-a su poboljšanje jezika sa podrškom za multimedije koji će biti razumljiv za ljude, ali isto tako i za kompjutere i uređaje (web pretraživače, parsere...). HTML5 je potencijalni kandidat za „cross-platform“ mobilne aplikacije pošto su mnoge osobine HTML5 pravljene tako da budu u mogućnosti da se pokrenu na uređajima male snage kao što su smart telefoni i tableti.

HTML5 je namenjen da prevaziđe ne samo HTML4, nego XHTML1 i HTML DOM. WHATWG je radio na web formama i aplikacijama, dok je W3C radio na XHTML 2.0 . 2006. godine su počeli zajedničku saradnju. Tako da se HTML5 može smatrati mešavinom karakteristika i specifikacija HTML-a i XHTML-a. Ovome je doprinela zajednička praksa, kao i mnoge greške u postojećim web dokumentima, ovo je takođe i pokušaj da se definiše jedinstveni markup jezik, koji se moţe pisati i u HTML-u i u XHTML-u. Ovo uključuje detaljne procesne modele da bi ohrabrilo još interoperabilnih implementacija. To proširuje, unapređuje i racionalizuje označavanje dokumenata, kao što je omogućen i API (application programming interfaces), za kompleksne web aplikacije. Trenutno postoje dve paralelne verzije HTML 5, jedna sluţbena na kojoj radi W3C i nesluţbenu na kojoj rade stručnjaci iz Applea, Mozille, Opere i Googlea, koji imaju velik uticaj na razvoj Interneta, iako iza sebe nemaju formalnu organizaciju kao što je W3C. Ipak, urednici HTML 5 specifikacije za oba tima su isti ljudi, što znači da grupe meĎusobno saraĎuju i da ćemo na kraju ipak imati jedan standard. Urednici HTML 5 specifikacije su Ian Hickson iz Googlea i David Hyatt koji radi za Apple.

HTML5 uvodi mnoge nove sintaksne promene. One uključuju nove <video>, <audio> i <canvas> elemente, kao i integraciju SVG (Scalable Vector Graphics) sadržaja (koji zamenjuju korišćenje <object> tagova). Ove karakteristike su dizajnirane tako da lako uključuju i rukuju multimedijalnim i grafičkim sadržajem na web-u bez potrebe uključivanja plugin-ova i API-ja. Drugi novi elementi, kao što su na primer <section>, <article>, <header> i <nav> su kreirani da obogate semantički sadržaj dokumenta. Sa druge strane, neki elementi i atributi su uklonjeni, kao na primer <font> i <center> koji su prevaziđeni korišćenjem mnogo moćnijeg css-a.

## 3.2.CSS

Iako koristi da se njime opisuju web stranice i korisnički interfejsi napisani u HTML-u i XHTML-u, CSS-om se takođe mogu opisati bilo koje vrste XML dokumenata uključujući i sam XML, SVG ili XUL. Uz korišćenje HTML i Javascript jezika, CSS daje vizuelnu strukturu i opis web sajtova, web aplikacija kao i aplikacija za mobilne telefone.

Tri osnovne karakteristike CSS jezika su mogućnost za definisanje klasa za izgled, boje i fontova. Ovi elementi omogućavaju pristupačniji i fleksibilniji sadržaj kao i kontrolu web dizajnera nad određenom grupom HTML elemenata u sadržaju. Na primer, znamo da HTML ima pojedine tagove kojima mogu da se definišu pojedini elementi, kao što je slučaj sa tagom <bold>. On omogućava podebljanost teksta na stranici. Ukoliko želimo da nam svaki naslov bude podebljan, korišćenjem CSS-a ćemo izbeći konstantno ponavljanje <bold> tagova na svakom mestu gde se nalazi naslov tako što ćemo definisati klasu za izgled <h1> tagova unutar CSS-a i samo pozvati tu klasu prilikom ispisivanja naslova.

CSS3 je najnoviji standard CSS-a. CSS3 specifikacija je i dalje u razvoju od strane W3C organizacije. Većina css3 svojstava je implementirana u modernim web pretraživačima.

CSS3 je podeljen na module. Stara specifikacija je podeljena na manje delove i dodati su novi delovi. Neki od najvažnijih modula(Box model, Backgrounds i borders, Tekstualni efekti i web fontovi,2D/3D transformacije, Animaceije, Multiple column layout, User interface, Selektori)

## 3.3.PHP

PHP je open source jezik koji se koristi za razvoj server-side aplikacija, kao i dinamičkog Web sadržaja. PHP dozvoljava interakciju sa velikim brojem relacionih baza podataka kao što su MySQL, Oracle, IBM D2, Microsoft SQL Server, PostgreSQL i SQLite. PHP radi na većini operativnih sistema današnjice, kao što su UNIX, Linux, Windows i Mac OS i može da interaguje sa većinom Web servera.

PHP je stekao popularnost zbog svoje jednostavnosti i sintakse nasleđene iz programskog jezika C. Tokom vremena jezik se proširivao i sticao mogućnosti za objektno orijentisano programiranje, naročito od verzije 5.0. Nalikuje jeziku C++ u smislu da dozvoljava i čisto-proceduralno programiranje ali omogućava i korišćenje klasa i drugih koncepata objektno orijentisanog programiranja (nasleđivanje, apstraktne metode, interfejsi itd.).

Neke od osnovnih odrednica PHP – a su:

* PHP je akronim za Hypertext Preprocessor
* server – side skriptni programski jezik
* skripte se izvršavaju na serveru
* podržava rad sa različitim bazama podatama (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, itd.)
* open source softver (OSS)
* radi na različitim platformama (Windows, Linux, Unix i td.)
* kompatibilan je sa skoro svim serverima koji se danas koriste (Apache, IIS, itd. )
* lak je za učenje i efikasno funkcioniše na strani servera.

Za PHP fajlove, bitne su sledeće činjenice:

* mogu da sadrže tekst, HTML tagove i skripte
* vraćaju se klijentskom browser – u kao HTML tekst (*plain HTML*)
* imaju ekstenziju: “.*php*”, “.*php3*” ili “.*phtml*”
* Program koji se napiše u PHP-u ne zahteva prevođenje (kompajliranje), nego se interpretira pri svakom izvršavanju. PHP interpreter može raditi po CGI principu, odnosno tako što će interpreter postojati kao eksterna aplikacija koja se poziva da izvrši datu skriptu svaki put kad bude zahtevana od nekog korisnika, a može biti instaliran i kao modul veb-servisa. Druga varijanta je danas u najvećoj upotrebi jer pruža znatno veću brzinu izvršavanja - interpreter je na taj način uvek učitan u memoriju te se ne mora pozivati spoljašnji program.

Uobičajen scenario po kom se izvršavaju PHP skripte na web serveru je sledeći:

* klijent (korisnik Interneta koji koristi neki brauzer) zahteva PHP stranicu sa servera
* server prosleđuje zahtev servisu za veb (program veb-server na serveru)
* veb-server prepoznaje da se traži PHP datoteka
* ne šalje njegov sadržaj klijentu, nego ga izvršava kao program pomoću PHP modula
* izlazni tekst programa (standardni izlaz) se šalje klijentu kao rezultat zahteva
* klijent prepoznaje vrstu rezultata (HTML kod, slika, PDF sadržaj, arhiva itd.)

rezultat se prikazuju klijentu na odgovarajući način

## 3.4.AJAX(Asynchronous JavaScript And XML)

AJAX je skraćenica od “Asynchronous JavaScript and XML”. Termin AJAX prvi put je upotrebljen u februaru 2005. godine, kada je Džesi Džejms Garet (Jesse James Garret), dizajner informacionih sistema i direktor kompanije Adaptive Path, pokušao da nađe odgovarajuću skraćenicu za grupu tehnologija koju je predlagao svom klijentu. Tehnologije potrebne za postojanje AJAX -a su postojale i ranijih godina ali je glavni razlog za slabiju upotrebu svih tehnika korišćenih u AJAX-u, nedostatak podrške od strane web čitača. Mnogi su godinama pre nje koristile slične principe, poput Microsoftovog Remote Scriptinga ili veoma raširenog DHTML-a. Najbitnija stvar koja je uticala na brz razvoj AJAX-a jeste trenutak na tržištu u kojem su veliki igrači Internet industrije želeli da težište korišćenja računara prenesu sa desktopa na web stranice, za šta im je bila potrebna upravo tehnologija.

Web aplikacije imaju brojne prednosti u odnosu na desktop aplikacije. Web aplikacije dosežu do većeg broja ljudi, lakše se razvijaju, implementiraju i održavaju.

Najveći nedostatak Web aplikacija u odnosu na desktop aplikacije je taj što su desktop aplikacije „bogatije” u smislu funkcija i informacija koje nude. Međutim, taj nedostatak je u velikoj meri otklonjen uz pomoć AJAX tehnologije.

Web aplikacije su one aplikacije čijom funkcionalnošću se upravlja preko web servera i dostavljaju se korisnicima preko mreže kao što je internet ili intranet. I pored mnogostrukih prednosti, klasične web aplikacije imaju određene neostatke, kada je u pitanju interakcija sa korisnicima:

* Spor odgovor na zahteve
* Gubitak sadržaja prilikom ponovnog učitavanja stranice
* Gubitak informacija na ekranu
* Gubitak pozicije skrolovanja
* Bez trenutnog odgovora na korisničke akcije
* Korisnik mora da čeka na učitavanje sledeće strane
* Klikni, čekaj, refresh princip rada na stranici
* Stranica se ponovno učitaca sa servera za sve akcije korisnika
* Sinhroni način rada.

Pre svega, svaki put kada se učitava nova stranica, postoji određeno vreme kada aplikacija «stoji». Veoma često se prilikom malih izmena ili zahteva za delovima stranice, ponovo učitava cela stranica, iako je najveći deo nove stranice u potpunosti identičan sa prethodnom.

Da bi se shvatio koncept AJAX tehnologija, neophodno je objasniti razliku izmenu sinhronog i asinhronog prenosa podataka između klijenta i servera.

U sinhronom načinu rada procesi se izvode sekvencijalno. Između dva procesa postoji jaz koji je jednak vremenu izvođenja drugog procesa. Dakle, osnovni proces komunikacije između klijenata i servera može se opisati na sledeći način: klijent šalje zahtev ka serveru, podaci se prenose ka serveru, server obrađuje podatke I zatim ih vraća klijentu. Ono što je primetno sa slike dole je da postoji vremenski interval između pravljenja zahteva i odgovora na isti. Za vreme tog intervala klijent čeka, bez mogućnosti da pravi nove zateve ili nastavi svoje korišćenje web aplikacije. Može se zaključiti da kod sinhronog prenosa podataka ne postoji mogućnost za eliminisanje čekanja zato što se procesi obavljaju jedan za drugim, čak i ukoliko za to nema potrebe.

## 3.5.JSON

JSON (JavaScript Object Notation) predstavlja tekstualni format za serijalizaciju podataka. Izveden je od formata za zapis objekta skriptnog jezika JavaScript, međutim danas ovaj format koristi većina savremenih programskih jezika. Osnovna svrha JSON-a je prenos podataka i može se reći da predstavlja svojevrsnu alternativu XML-u. JSON je u memorijskom smislu manje zahtevan od XML-a, što znači da JSON fajl sa određenim podacima zauzima manje memorijskog prostora od XML fajla sa istim podacima.

JSON je zasnovan na JavaScript standardu Standard ECMA-262 3rd Edition - December 1999. Iako je JSON usko povezan sa JavaScript programskim jezikom, danas praktično svi programski jezici podržavaju JSON zapis podataka. Među njima su JavaScript, PHP, Java, C, C++, Pyhon, Ruby i drugi.

JSON se zasniva na dve strukture:

* Zbirka parova ime / vrednost. Na raznim jezicima, to je realizovano kao objekat, zapis, struktura, rečnik, heš tabela, lista sa ključevima ili asocijativni niz.
* Uređena lista vrednosti. U većini jezika, ovo je realizovana kao niz, vektor, lista ili sekvenca.

JSON format razlikuje više tipova podataka. Promenljive se u JSON-u nazivaju vrednosti. Od prostih tipova podataka, to su string, number, boolean i null, a od složenih tipova u pitanju su object i array.

## 3.6.JavaScript

JavaScript je objektno zasnovan skriptni jezik. Uključujemo ga u web stranicu da bi je učinili dinamičnijom. HTML (osnovni kod web stranica) se koristi samo za oblikovanje i uređivanje elemenata stranice (tekst, forme, linkove i tabele), ali nema šanse da diktiramo kako će se ti elementi ponašati. Mogućnost uključenja JavaScript skripte daje nam mnogo veću kontrolu kako se web stranica ponaša. Kombinovan sa HTML-om i CSS-om JavaScript čini DHTML (Dinamic HTML).

Objektno je zasnovan jer programer ne definiše samo tip podataka, nego i vrstu operacija (funkcija) koje se mogu primjeniti na strukture podataka. Na ovaj način, struktura podataka postaje objekat koji uključuje i podatke i funkcije. Pored toga, programeri mogu da kreiraju odnose između jednog i drugog objekta. Na primjer, objekti mogu da pridobiju karakteristike od drugih objekata.

Skriptni je jezik jer se sastoji od serije komandi koje se očitavaju u interpreteru (program prevodioc), a da se predhodno ne kompajlira sadržaj (compiler- program prevodioc). Odnosno ne prevodi se u mašinski jezik (binarni kod- 1 i 0) iz koga nikada nećemo saznati originalni jezik, nego se komande direktno "čitaju" iz koda (source code ili bytecode). Zbog ove karakteristike JavaScript se izvršava na strani korisnika (client side), tj. na računaru na kojem je pokrenut sadržaj sa JavaScript-om. Sam po sebi, HTML dozvoljava posjetocu da pošalje podatke ka serveru na obradu. Nažalost ako ti podaci nisu validni cijeli proces se mora ponoviti sve dok se ne unesu validni podaci. Ovo je jedan od osnovnih razloga nastanka JavaScript-a koji provjerava vjerodostojnost podataka na klijentovom pregledniku (browser-u) i tako olakšava posao na web-u.

Rani skript jezici su se često nazivali batch jezici. Ovo su neki skrip jezici : ASP, JSP, PHP, Perl, Tcl, Python itd.

JavaScript je najpopularniji skriptni jezik na Internetu kojeg podržavaju svi poznatiji preglednici (Internet Explorer, Mozilla Firefox, Netscape, Opera). Evo nekoliko stvari koje je svaki prosječan surfer na internetu vidio: padajući meni, neobični pokazivač miša, iskakajući prozor, sat ... Međutim postoje ozbiljnije primjene JavaScript-a kao što su:

Detekcija preglednika -Određuje se tip preglednika koji se koristi pri pregledu vaše web stranice. Zavisno od preglednika, može se prikazati drugačija stranica posebno dizajnirana za taj preglednik.

Kontrola prozora -mogu se kontrolisati dimenzije, meniji, dugmadi, vrijeme prikaza itd.

Web kolačići (cookies) -pohranjeni podataci na korisnikovom računaru, koji se automatski preuzimaju sljedeci put kada korisnik posjeti vašu web stranicu.

## 3.7.JQuery

jQuery je biblioteka čija je svrha da olakša upotrebu JavaScript-a na vašim web stranicama.

jQuery slogan „write less, do more“ dosta govori – sa jednom linijom jQuery koda možete pozvati funkcionalnosti za koje bi vam trebalo mnogo više rada i truda, ako bi koristili klasični JavaScript metod kodiranja.

jQuery obuhvata mnoge zadatke koji zahtevaju veliki broj linija JavaScript-a i omotava ih u metode koje možete pozvati sa samo jednom linijom.

jQuery takođe pojednostavljuje i druge komplikovane stvari u JavaScript-u kao što su AJAX i DOM manipulacija.

JQUERY BIBLIOTEKA SADRŽI SLEDEĆE FUNKCIONALNOSTI:

* HTML/DOM manipulacija
* CSS manipulacija
* HTML event metodi
* Efekti i animacije
* AJAX

**Dodatno:**[jQuery poseduje plugin-e](http://www.popwebdesign.net/popart_blog/2013/08/najbolji-jquery-dodaci/) sa kojima možete postići gotovo svaki zahtev koji zamislite.

Najvažniji razlog tolike poularnosti jQuery biblioteke je ekstezivnost. Veliki broj najvećih svetskih kompanija kao što su Google, Microsoft, IBM ga koriste.

Tri stvari su značajne i izdvojile su jQuery u odnosu na druge konkurente:

* korišćenje CSS selektora
* unobtrusive (nenametljiv) JavaScript
* jednostavnost upotrebe

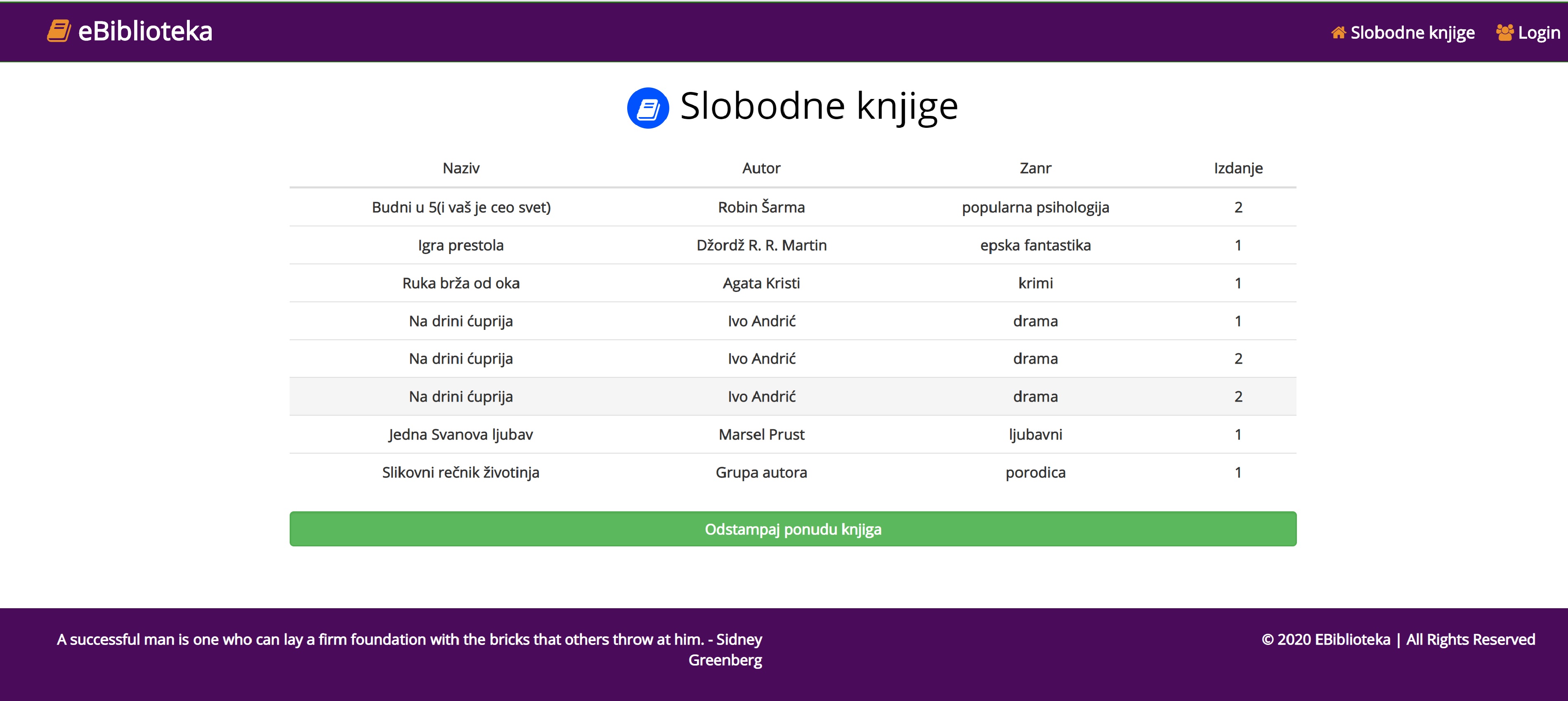
jQuery se pojavio u vreme kada su se front-end developeri navikavali na kreiranje web stranica sa boljim semantičkim obeležavanjem, a učenje CSS-a je bilo zastupljeno širom web-a. jQuery koristi potpuno iste [CSS selektore](http://www.popwebdesign.net/popart_blog/2013/08/10-korisnih-css-selektora/) koje developeri koriste za stilizovanje web stranica da bi dodali ponašanje i nema potrebe za promenom konteksta kada se referencira na iste elemente. „Unobtrusive JavaScript“je metodologija koja definiše napredan način inplementiranja JavaScript koda na web stranicama. Suština ovog pristupa je u tome da se odvaja funkcionalnost (sloj ponašanja) od strukture/sadržaja i prezentacije.Na ovaj način izbegavaju se standardni problemi JavaScript kodiranja kao što su nekonzistentnost na browser-ima i nedostatak skalabilnosti.

jQuery je napravljen tako da bude krajnje jednostavan za developere koji ga koriste

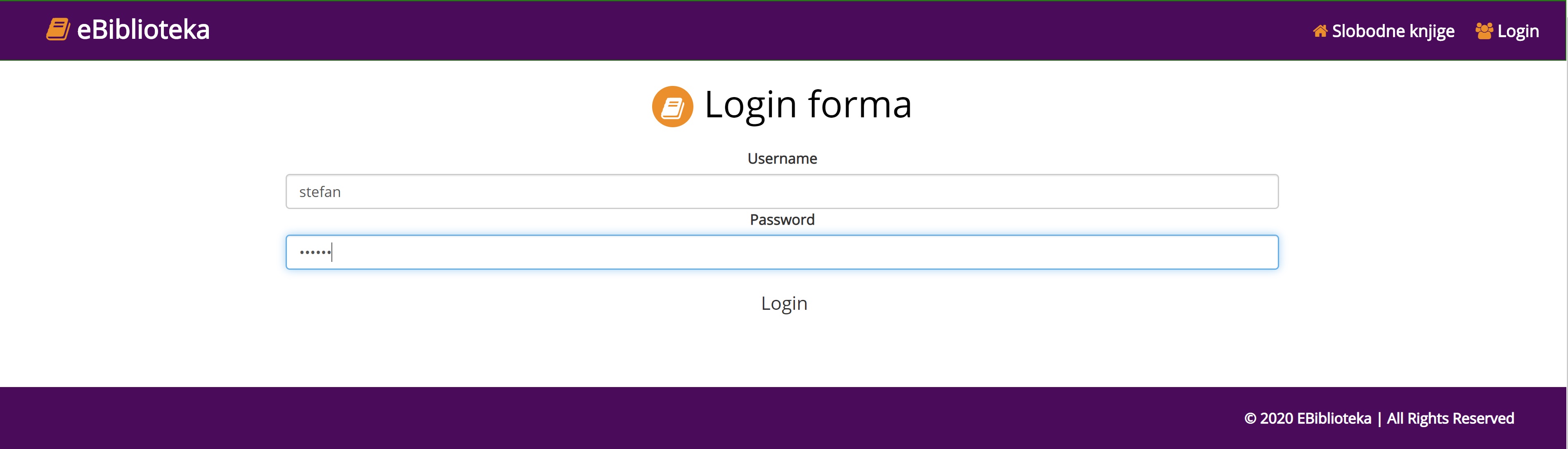
# 4.Korisničko uputstvo - Opis karakterističnih slučajeva korišćenja preko screenshot-ova aplikacije

## 4.1.Početna strana(Slučaj kada niko nije ulogovan)

Pri inicijalnom otvaranju sajta bibliotekari dobiju prikaz početne strane na kojoj se prikazuje spisak slobodnih knjiga koje je moguće formatizovati u PDF format prilikom odabira opcije „Odštampaj ponudu knjiga“. Takođe je prikazana i opcija „Login“ putem koje bibliotekari mogu da se prijave na svoj profil.



Slika1 : Početna strana

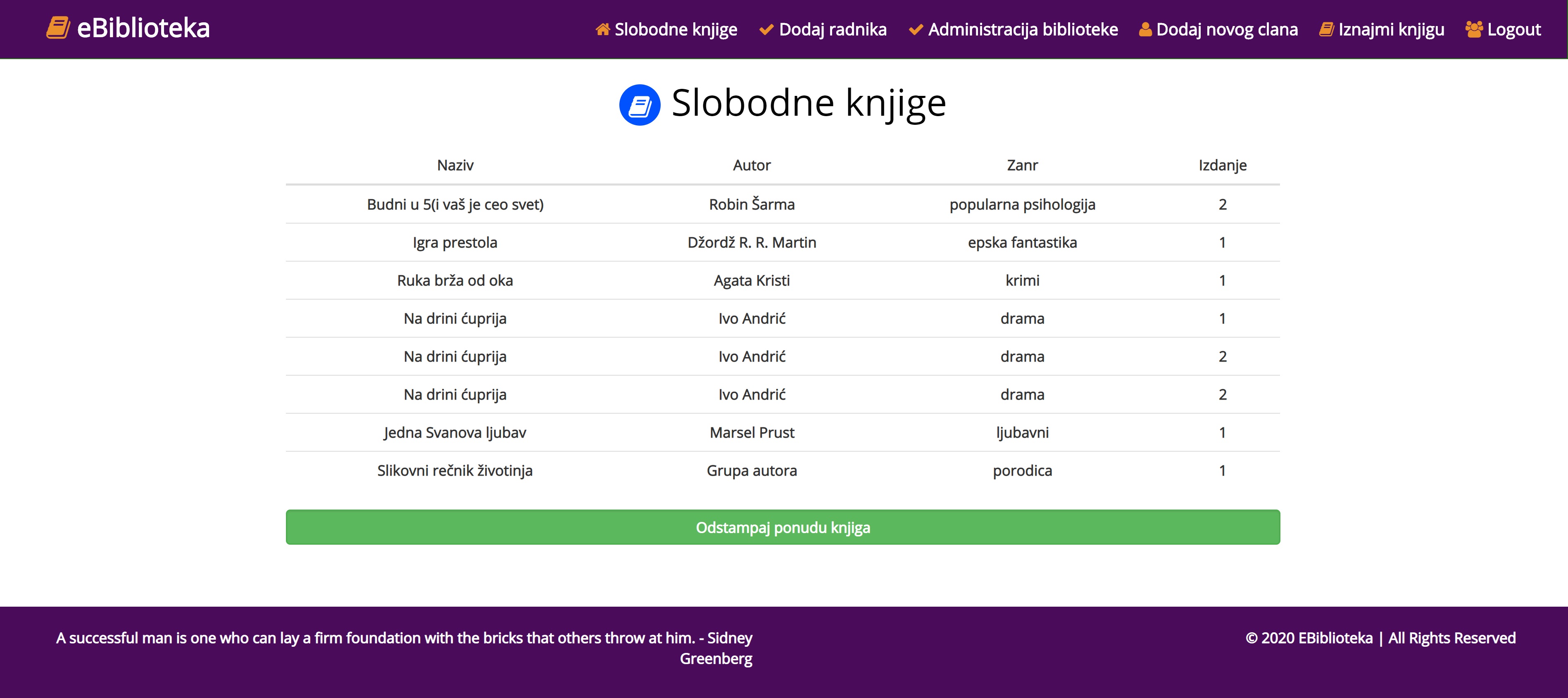


*Slika2 : Opcija „Login“*

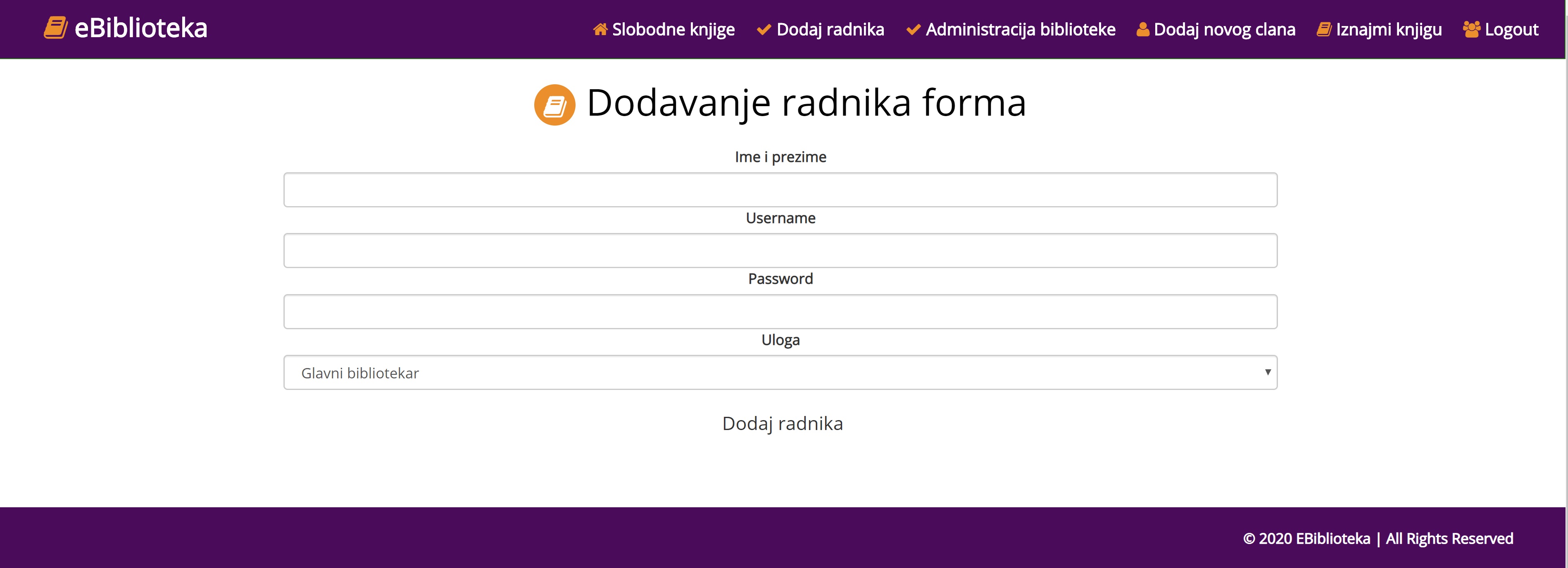
Na Kartici „Login“ bibliotekari unose svoje korisničko ime i šifru kako bi se prijavili na svoj profil.

## 4.2. Slučaj kada je ulogovan Glavni bibliotekar

Početna strana kada je na sajtu prijavljen glavni bibliotekar ima takođe prikaz liste slobodnih knjiga koje je moguće formatizovati u PDF fajl za štampanje, ali i dodatne opcije „Dodaj radnika“, „Administracija biblioteke“, „Dodaj novog člana“ i opcija „Iznajmi knjigu“ i „Logout“ - putem koje se glavni bibliotekar odjavljuje sa svog profila.



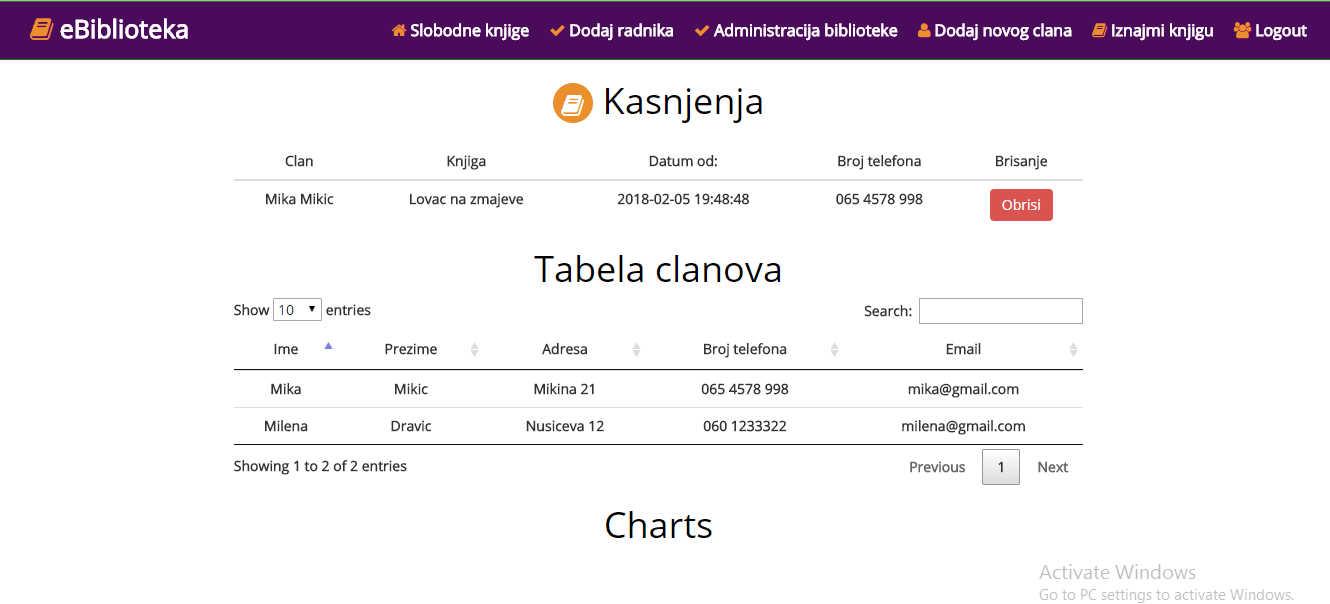
Slika - Početna stana(Kada je prijavljen glavni bibliotekar)



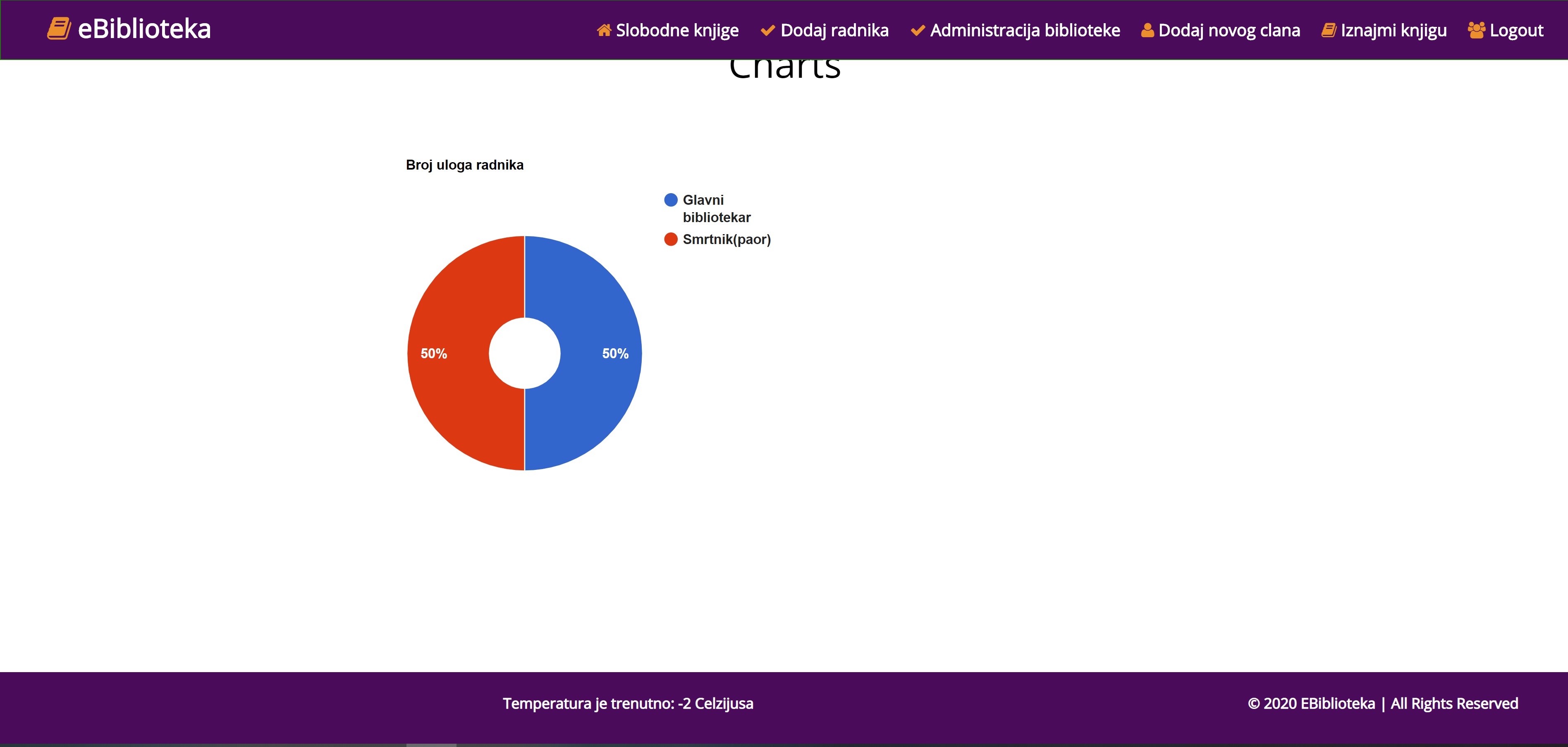
Slika - Opcija „Dodavanje radnika“

U kartici „Dodavanje radnika“, glavni bibliotekar pravi profil novog radnika. Potrebne informacije za unos novog radnika(Ime i prezime, username, password i ulogu koju ce radnik imati).

Pri uspešnom unosu ispisuje se poruka o uspesnom unošenju, u suprotnom se ispisuje poruka o neuspešnom unošenju :).



Slika -Opcija „Administracija biblioteke“-part1



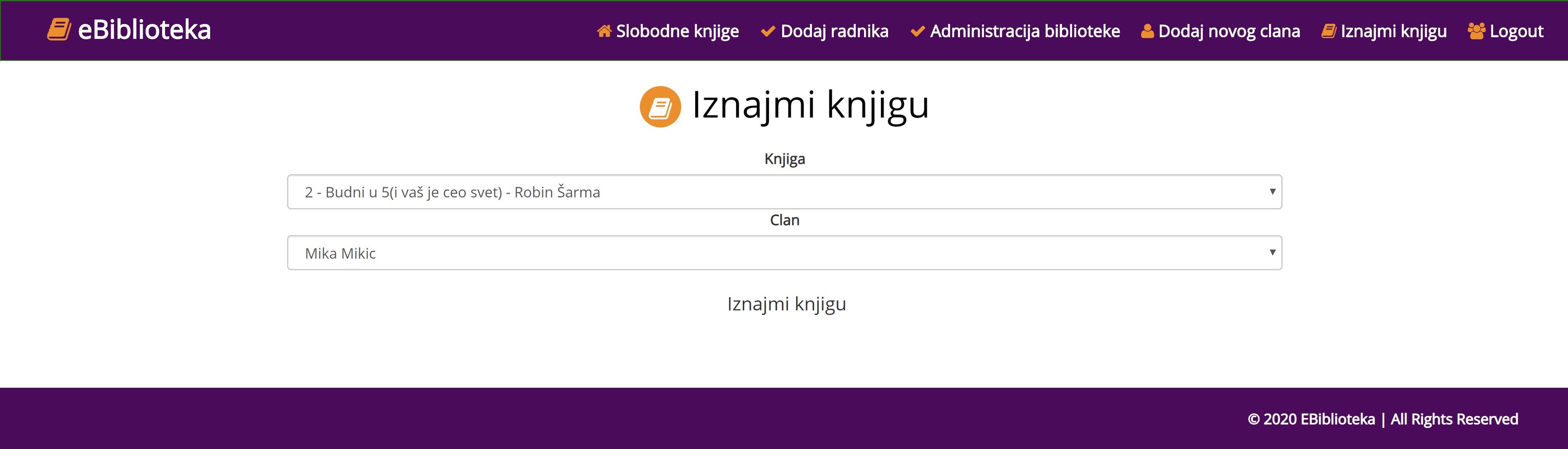
Slika - Opcija „Administracija biblioteke“-part 2

Kartica „Administracija biblioteke“ prikazuje listu članova koji kasne sa vraćanjem iznajmljene knjige, listu svih članova, i chart u kojem je prikazan procentualni odnos uloga zaposlenih.



Slika – Kartica „Dodavanje člana“

U kartici „Dodavanje člana“ , slično kao u kartici „Dodavanje radnika“ unose se potrebne informacije za dodavanje novog člana(Ime, Prezime, Email, Broj telefona, Adresa).

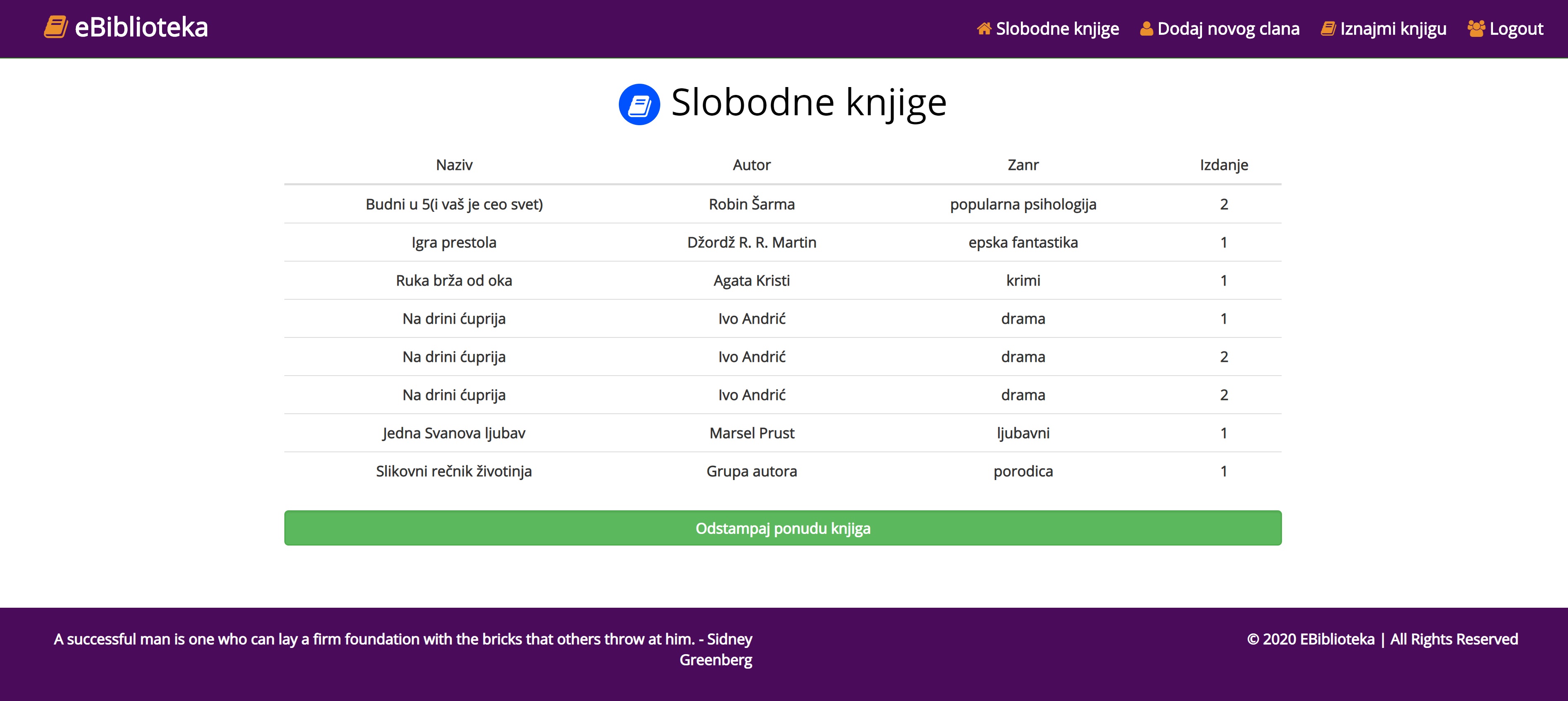


Slika - Kartica „Iznajmi knjigu“

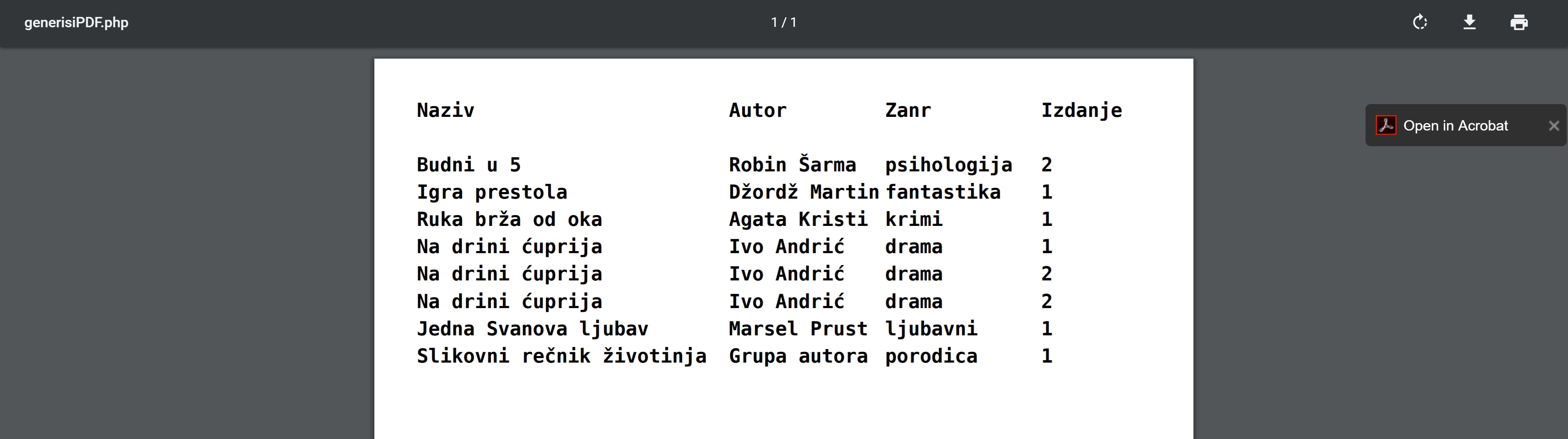
Preko kartice „Iznajmi knjigu“ glavni bibliotekar unosi informacije koju knjugu zadužujei kog člana zadužuje.

## 4.3. Slučaj kada je na sajtu prijavljen paor

Početna strana kada je prijavljen paor pored liste slobodnih knjiga ima i opcije „Dodaj novog člana“, „Iznajmi knjigu“ i „Logout“ koje su identične kao i kada je ulogovan glavni bibliotekar.



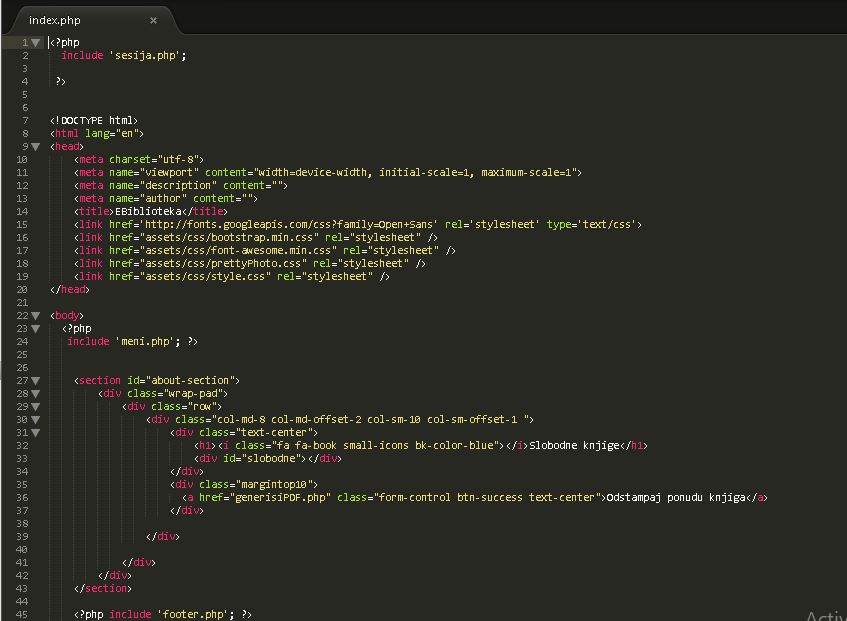
Slika 7 -Početna strana kad je prijiavljen paor



Slika 8 - Rezultat funkcije Odštampaj ponudu sa početne strane

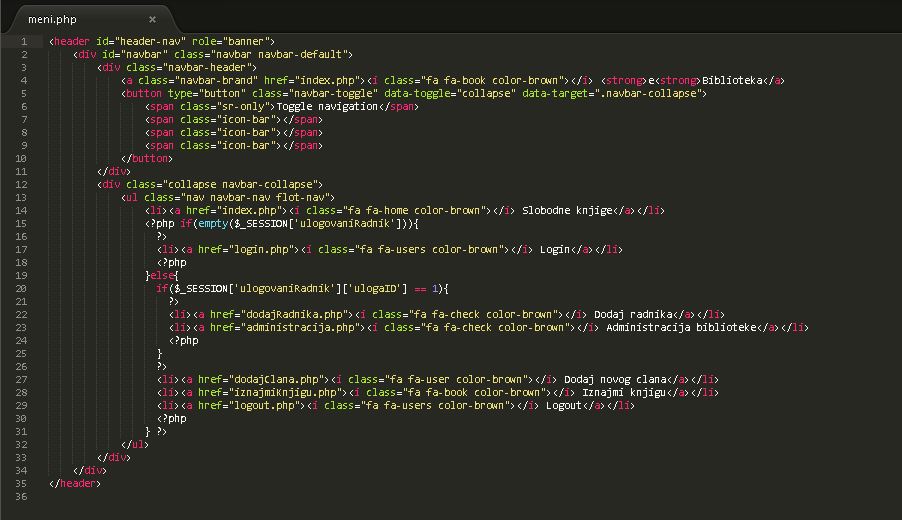
# 5.Prikaz reprezentativnih delova koda

U okviru ovog poglavlja, prikazani su neki delovi koda naše aplikacije elektronska biblioteka.



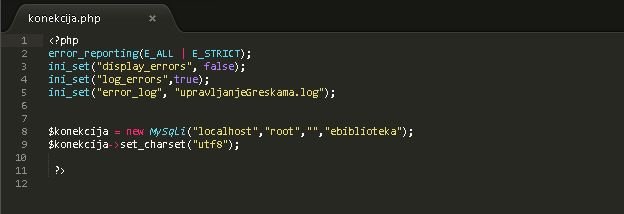
Izgled koda 1 – index.php

Pomoću funkcije include uključili smo fajlove meni.php i footer.php, kako se ne bi iznova ponavljali u okviru našeg koda.



Izgled koda 2 – Meni.php

U okviru menija, stranice menija su raspoložive u odnosu na to koji zaposleni radnik elektronske biblioteke je ulogovan u aplikaciji. Konkretno, ukoliko je u okviru sesije ulogovan radnik sa ID-em 1 što je naš primarni ključ za glavnog bibliotekara, automatski dobija dodatne stranice u odnosu na običnog paora.

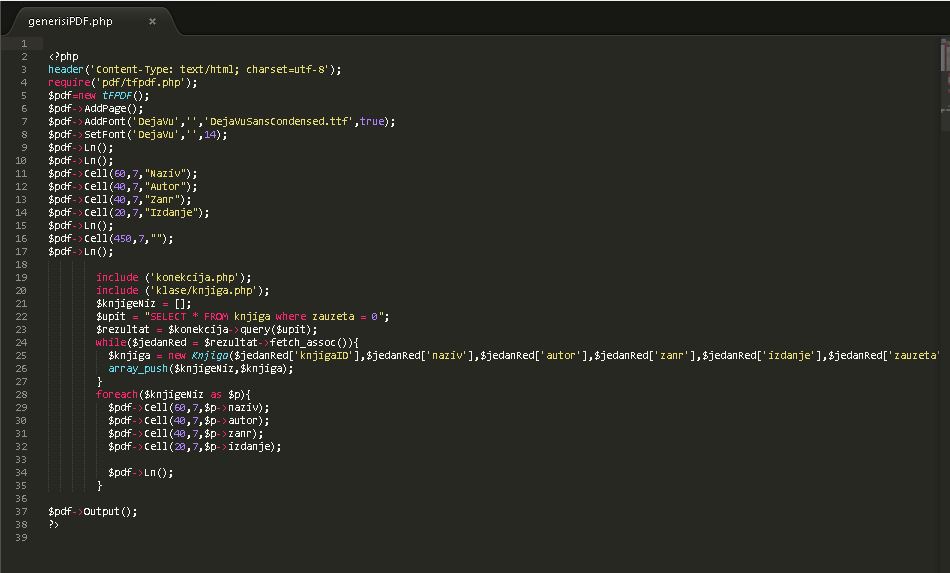


Izgled koda 3 – konekcija.php

Svaka funkcionalna veb aplikacija koristi neku bazu podataka, stoga je neophodno da je povežemo sa njom.

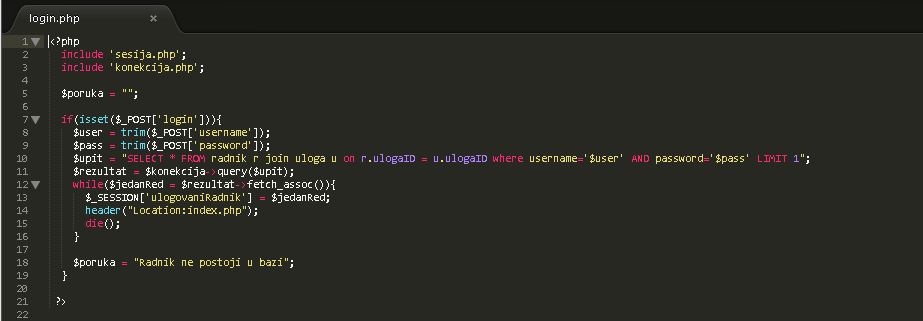
U prvih četiri koraka prosleđujemo parametre neophodne za konekciju, a to su server, user, password i naziv baze ebiblioteka. Vrši se povezivanje preko funkcije mysqli u kojoj prosleđujemo unete parametre.

Ako se konekcija neuspešno izvrši ispisuje se poruka o grešci.



Izgled koda 4 – generisiPDF.php

Koristili smo dodatne klase za definisanje pdf-a, poput tfpdf.php koju smo ubacili putem funkcije require. Da bismo došli do informacija o slobodnim knjigama, neophodno je da prvo uključimo konekcija.php i klasu knjiga.php i potom rezultat postaju oni objekti gde je zauzeta = 0. Svaka definisana ćelija pdf-a, će biti neki od atributa objekta knjiga.



Izgled koda 5 – login.php

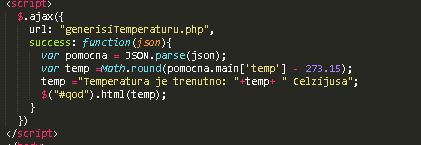
Bitnost koda iznad se ogleda u tome što smo za prenos podataka pri loginu koristili funkciju POST koja ne prikazuje parametre kroz komandnu liniju već se šalju transparentno kroz telo http zahteva.

Ukoliko je zaposlen uspešno ulogovan, biće preusmeren na index.php stranicu. Ukoliko zaposlen ne postoji, biće prikazana poruka da on ne postoji u bazi.



Izgled koda 6 – administracija.php

U okviru ove funkcije, u tabelu se ubacuju svi članovi koji kasne sa vraćanjem knjige u biblioteku.



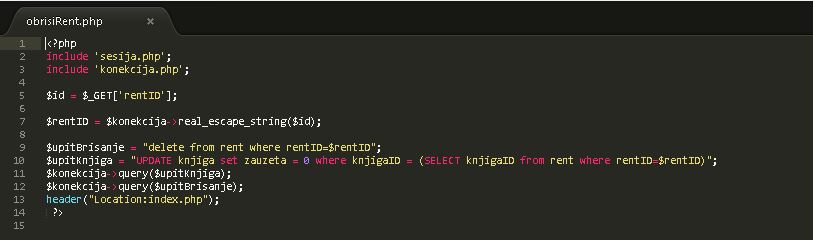
Izgled koda 7 – administracija.php

U okviru našeg futera, na svakoj stranici moguće je učitavanje temperature.



Izgled koda 8 – administracija.php

Formatiran je uobičajeni google forms izgled dijagrama „pie“ i broj radnika u konkretnoj ulozi je prikazan preko broja pojavljivanja, ali procentualno izražen.



Izgled koda 9 – obrisiRent.php

Putem ove funkcije, knjiga koja je vraćena u biblioteku opet postaje slobodna u komunikaciji sa bazom.



Izgled koda 10 – dodajClana.php

Putem POST metode koja je karakteristična za veb servise, možemo u bazu dodati novog člana naše biblioteke. Slično je i za dodavanje novog radnika za šta je privilegovan samo glavni bibliotekar.



Izgled koda 11 – iznajmiKnjigu.php

Nakon što smo u tabelu rent ubacili konkretnu knjigu i člana koji ju je iznajmio, potrebno je da se u tabelu knjiga njena vrednost zauzeta postavi na 1 i kao krajnji rezultat korisnik dobija poruku – „Uspesno ste iznajmili knjigu“ ili ako ne –„ Neuspesno iznajmljivanje“.