Univerzitet u Beogradu

Fakultet organizacionih nauka

Laboratorija za elektronsko poslovanje



APLIKACIJA ZA STUDENTSKU SLUŽBU

Seminarski rad iz predmeta Internet tehnologije

Nastavnik: Doc. dr Dušan Barać  
  
  
Student: Uroš Nedeljković 0234/2014

Beograd, 2019.

Sadržaj

[1 Korisnički zahtev 3](#_Toc506683547)

[2 Slučajevi korišćenja 3](#_Toc506683548)

[2.1 Prijava na sistem 4](#_Toc506683549)

[2.2 Upis studenata 5](#_Toc506683550)

[2.3 Unos rasporeda ispita 5](#_Toc506683551)

[2.4 Prijava ispita 5](#_Toc506683552)

[2.5 Unos ocena 5](#_Toc506683553)

[2.6 Pregled korisnika 5](#_Toc506683554)

[3 Arhitektura 10](#_Toc506683555)

[3.1 Dijagram klasa 10](#_Toc506683556)

[4 Specifikacija REST API-ja 11](#_Toc506683557)

[5 Opis korišćenih tehnologija 11](#_Toc506683558)

[HTML 11](#_Toc506683559)

[CSS 12](#_Toc506683560)

[PHP 12](#_Toc506683561)

[AJAX(Asynchronous JavaScript And XML) 13](#_Toc506683562)

[JSON 14](#_Toc506683563)

[JavaScript 14](#_Toc506683564)

[Flight 15](#_Toc506683565)

[6 Korisničko uputstvo 15](#_Toc506683566)

[6.1 Službenik 16](#_Toc506683567)

[6.2 Administrator 18](#_Toc506683568)

# Korisnički zahtev

Potrebno je projektovati i implementirati veb aplikaciju poslovnog informacionog sistema studentske službe Fakulteta organizacionih nauka. Sistem treba da radi sa tri vrste korisnika: sa studentima, zaposlenima u studentskoj službi i sa administratorima sistema. Korisnicima sistema treba omogućiti logovanje na sistem. Studentima omogućiti prijavu ispita. Studentska služba zadužena je za unos/izmenu ocena,unos studenata u sistem, kao i dodavanje predmeta. Administrator sistema ima pristup istim funkcionalnostima kao i zaposleni u studentskoj službi, a osim tih funkcionalnosti ima mogućnost izmene.

Slučajevi korišćenja:

1. Prijava korisnika na sistem

2. Upis studenata

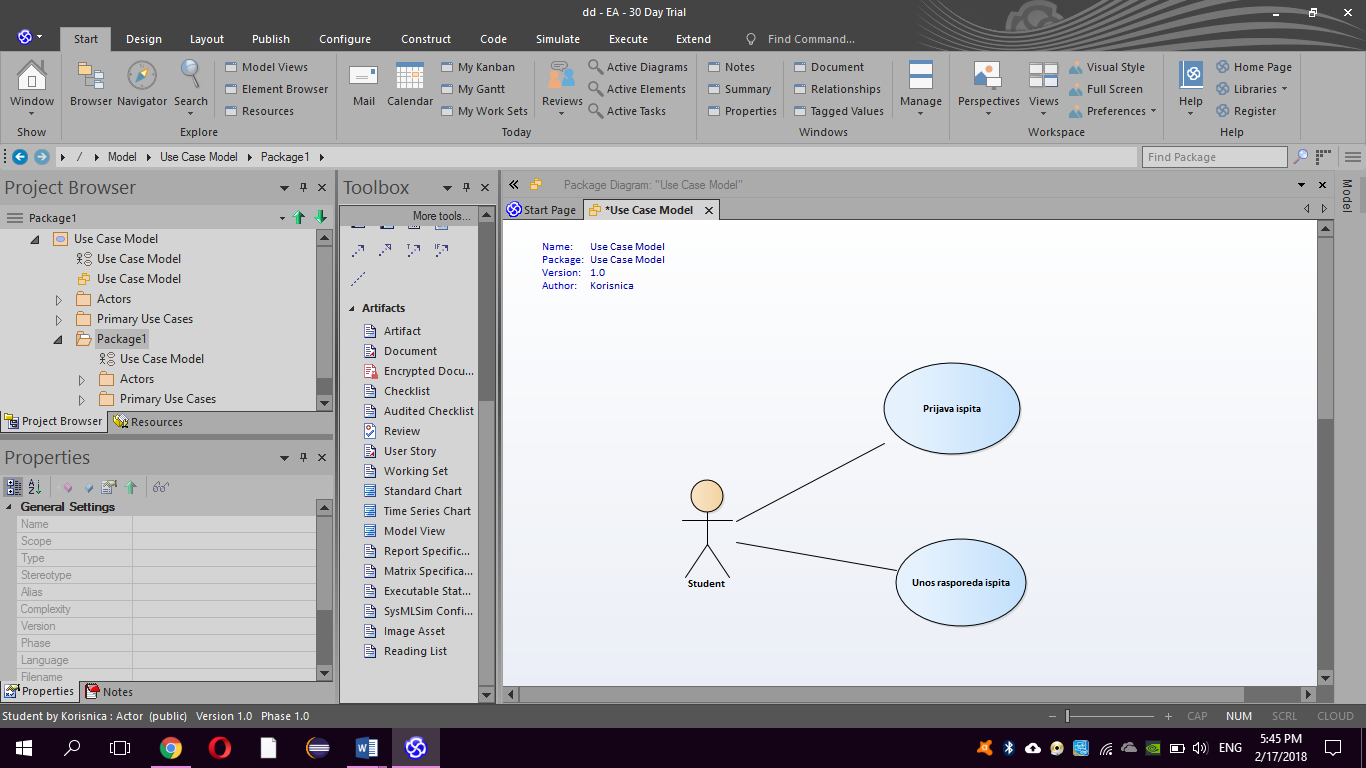
3. Unos rasporeda ispita

4. Prijava ispita

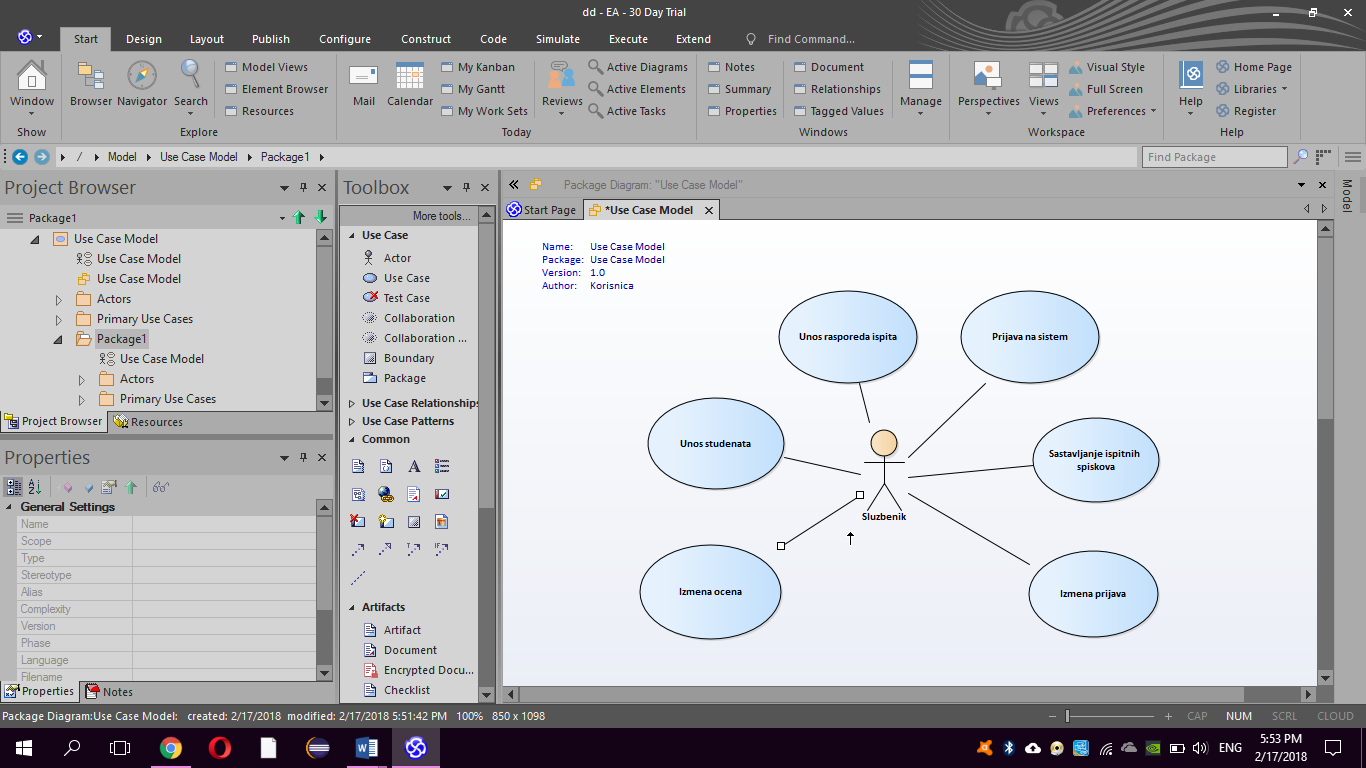
5.Unos ocena

6. Pregled korisnika

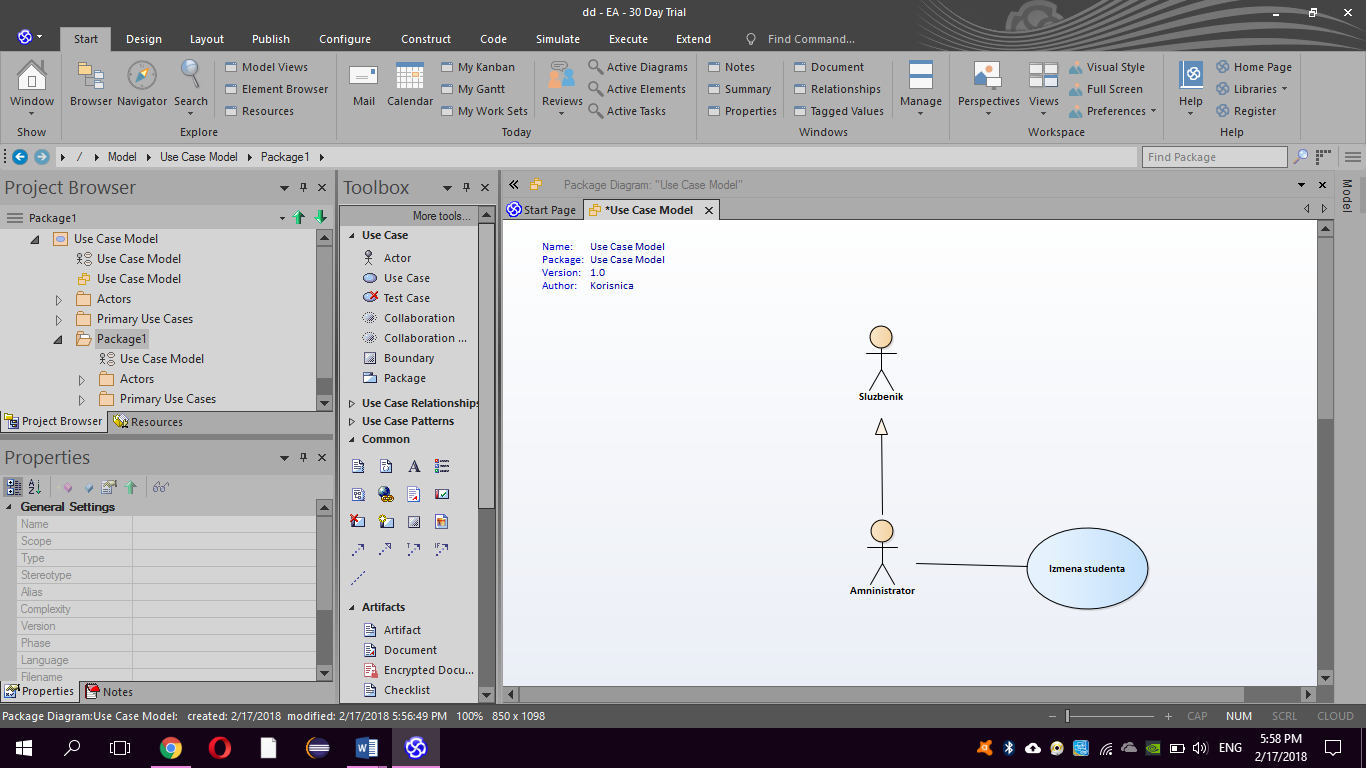
# Slučajevi korišćenja



Slika 1: Dijagram slučajeva korišćenja za studente



Slika 2: Dijagram slučajeva korišćenja za zaposlene u studentskoj službi



Slika 3: Dijagram slučajeva korišćenja za administratore

## Prijava na sistem

U vreme ispitnih rokova, studenti popunom ispitnih prijava prijavljuju one predmete koje žele da

polažu u definisanom ispitnom roku. Sve prijave se evidentiraju u studentskoj službi koja proverava da li su uneti podaci na prijavi korektni i da li student ima pravo da polaže prijavljeni ispit.

Vanredni studenti plaćaju svaku prijavu ispita bez obzira da li su izašli na ispit ili ne.Uvidom u dosije, a ako je broj pokušaja polaganja ispita veći od tri tj pet, student dodatno plaća odgovarajuću nadoknadu.

Ako student ne podnese prijavu na vreme, naknadni prijem ispitne prijave se takođe plaća.

Na osnovu ispravnih prijava formiraju se ispitni spiskovi koji se dostavljaju predmetnim nastavnicima, dok se studenti o vremenu i grupi za polaganje obaveštavaju spiskom na oglasnoj tabli. Način polaganja ispita je definisan postojećim nastavnim planom i programom.

## Upis studenata

Na ispitu, nastavnik za svakog prijavljenog studenta koji je izašao da polaže ispit popunjava u ispitnoj

prijavi ocenu koju je student dobio. Za ostale se evidentira da student nije izašao na ispit. Rezultati

ispita i ispitne prijave se dostavljaju studentskoj službi, koja sve ispitne prijave raspoređuje po

odgovarajućim dosijeima prijavljenih studenata (nezavisno od toga da li je student položio ispit ili ne,

odnosno da li je uopšte izašao na ispit).

## Unos rasporeda ispita

Pre početka ispitnog roka, službenik unosi raspored ispita u sistem.Ukoliko je ispitni rok u periodu održavanja prijemnog ispita i upisivanja studenata, prvobitan raspored je određen intervalima, nakon čega se određuju tačni datumi ispita, te službenik ažurira raspored ispita.

## Prijava ispita

U vreme ispitnih rokova, studenti popunom ispitnih prijava prijavljuju one predmete koje žele da polažu u definisanom ispitnom roku. Sve prijave se evidentiraju u studentskoj službi koja proverava da li su uneti podaci na prijavi korektni i da li student ima pravo da polaže prijavljeni ispit.

Vanredni studenti plaćaju svaku prijavu ispita bez obzira da li su izašli na ispit ili ne.Uvidom u dosije, a ako je broj pokušaja polaganja ispita veći od tri tj pet, student dodatno plaća odgovarajucu nadoknadu.

Ako student ne podnese prijavu na vreme, naknadni prijem ispitne prijave se takođe plaća.

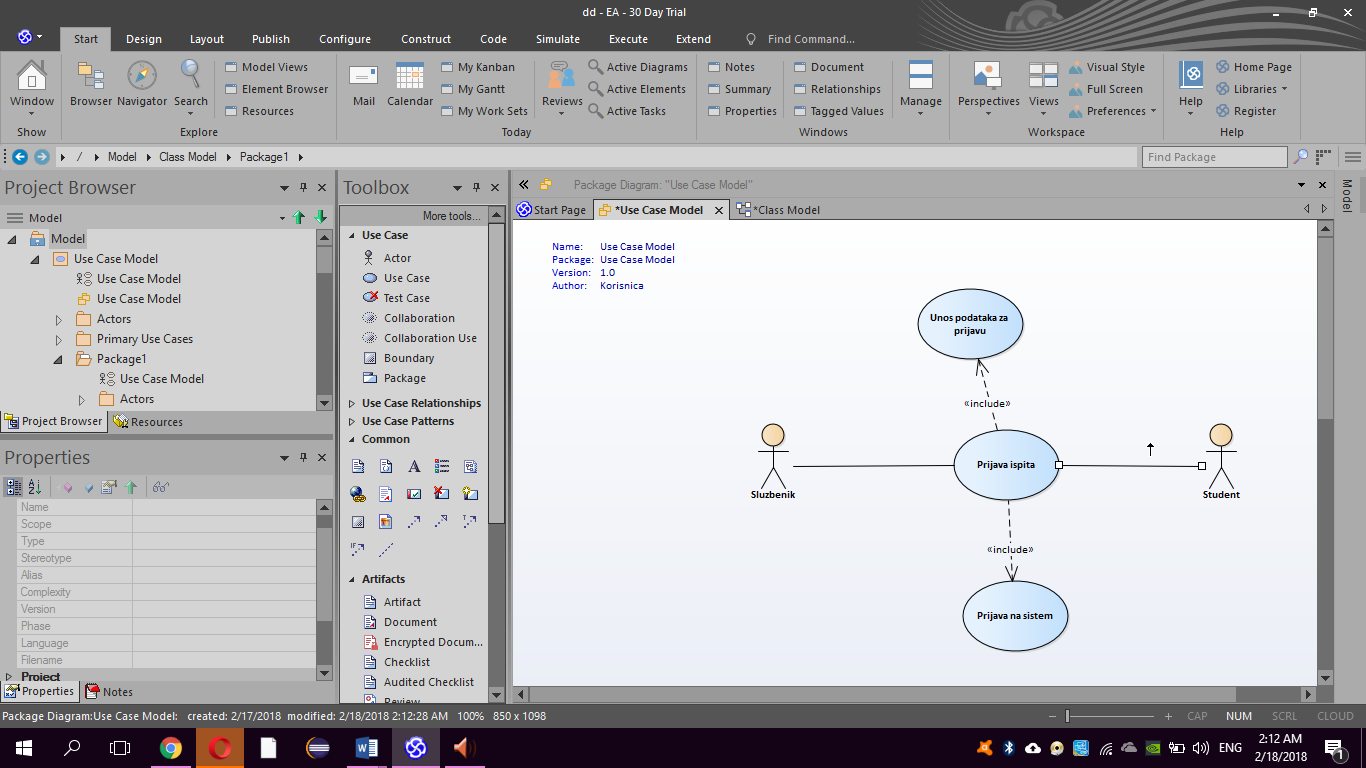
Na osnovu ispravnih prijava formiraju se ispitni spiskovi koji se dostavljaju predmetnim nastavnicima, dok se studenti o vremenu i grupi za polaganje obaveštavaju spiskom na oglasnoj tabli. Način polaganja ispita je definisan postojećim nastavnim planom i programom.

## Unos ocena

Službenik, nakon završetka ispitnog roka unosi ocene na sistemu.Po default-u ocena je 5,te ukoliko student nije izašao na ispit ili nije položio ocena ostaje nepromenjena. Takođe, mogući su slučajevi grešaka, te službenik vrši ažuriranje ocene.

## Pregled korisnika

Na sistemu studentske službe omogućen je celokupan pregled korisnika, odnosno, studenta , što znači da je moguće videti ispite koje je polagao,koje ocene ima kao i status školovanja, i lične podatke korisnika.



Slika 4: Slučaj korišćenja – Prijava ispita

**Slučaj korišćenja – Prijava ispita**

***Naziv SK:*** Prijava ispita

***Aktori SK:*** Zaposleni u službi, student

***Učesnici SK:*** Zaposleni u službi, student i sistem

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

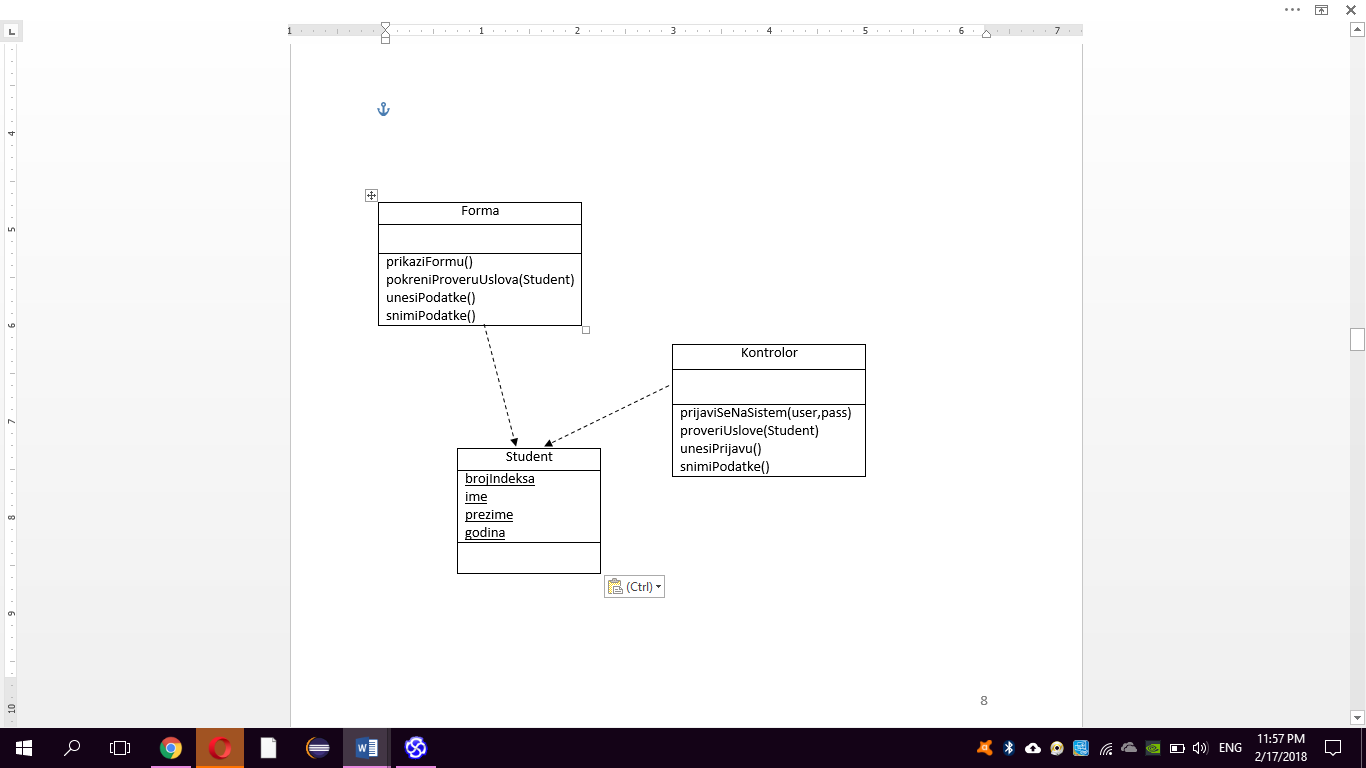
***Osnovni scenario:***

1. Zaposleni u službi se prijavljuje na sistem - slučaj korišćenja 1. Prijava na sistem
2. Zaposleni u službi saopštava sistemu da želi da pokrene unos podataka o prijavi ispita studenta studenta. (APSO)
3. Sistem prikazuje formu za unos podataka o prijavi ispita. (IA)
4. Zaposleni u službi saopštava sistemu da želi da pokrene proveru zadovoljenja uslova za prijavu ispita. (APSO)
5. Sistem obaveštava zaposlenog u službi da su uslovi zadovoljeni. (IA)
6. Zaposleni u službi unosi podatke u sistem o prijavi ispita
7. Zaposleni u službi predaje podatke sistemu. (APSO)
8. Sistem obaveštava zaposlenog u službi da je prijava ispita uspešno izvršena (IA).

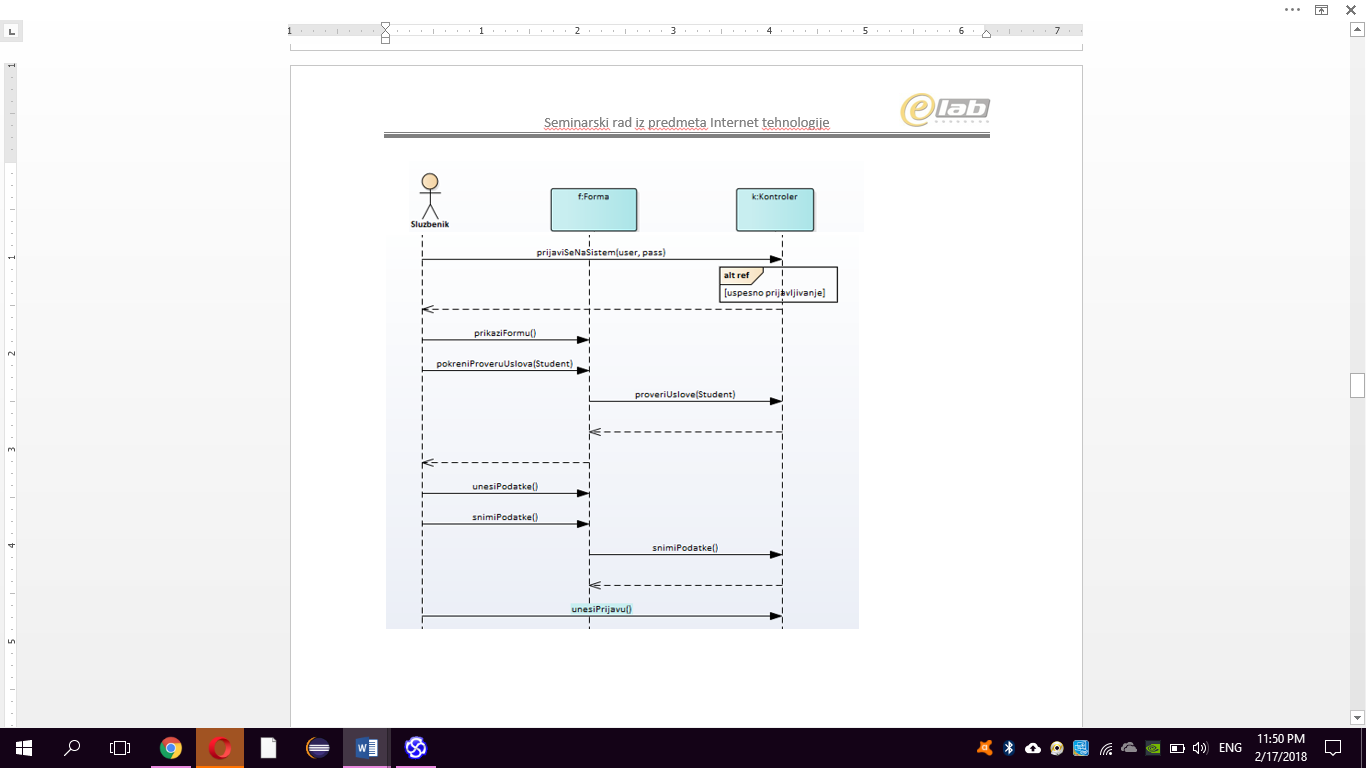
***Alternativni scenario:***

5.1 Sistem saopštava zaposlenom u službi da uslov prijave ispita nije ispunjen. (IA)

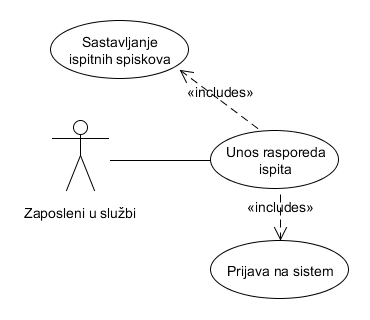
8.1 Sistem saopštava zaposlenom u službi da prijava ispita nije uspešno izvršena. (IA)



Slika 5: Dijagram klasa – prijava ispita



Slika 6: Dijagram sekvence – prijava ispita



Slika 7: Slučaj korišćenja - Sastavljanje ispitnih spiskova

**Slučaj korišćenja – Unos rasporeda ispita**

***Naziv SK:*** Sastavljanje ispitinih rokova

***Aktori SK:*** Zaposleni u službi

***Učesnici SK:*** Zaposleni u službi i sistem

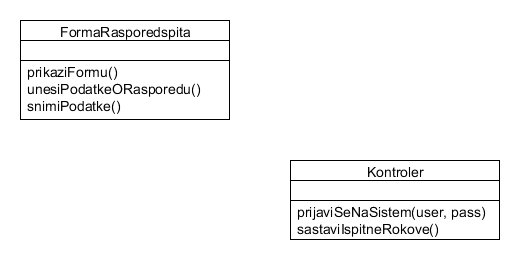
***Preduslov:*** Sistem je pokrenut.

***Osnovni scenario:***

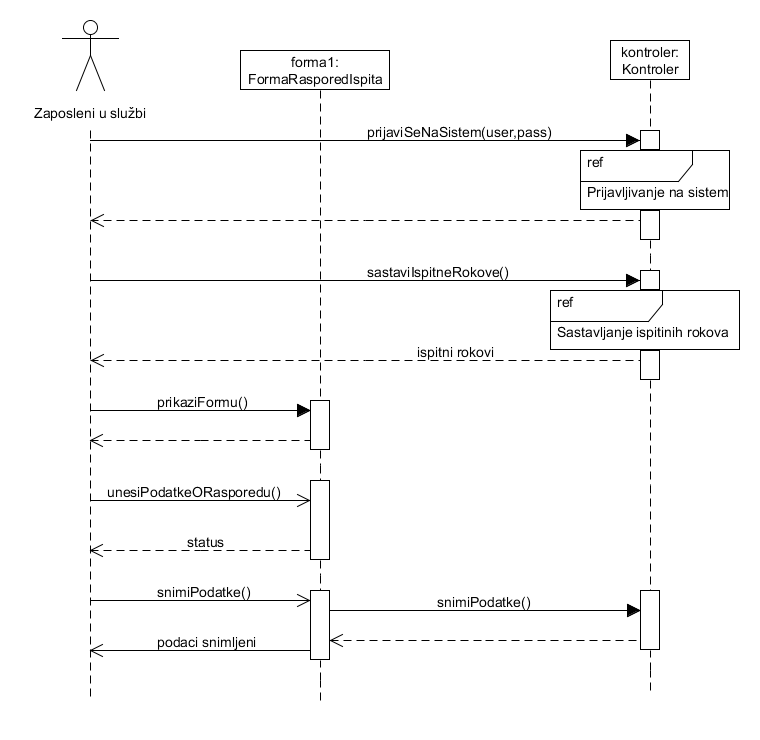
1. Zaposleni u službi se prijavljuje na sistem — slučaj korišćenja 1. Prijava na sistem
2. Zaposleni u službi sastavlja ispitne spiskove — slučaj korišćenja 6. Sastavljanje ispitnih spiskova
3. Zaposleni u službi saopštava sistemu da želi pokrene unos rasporeda ispita. (APSO)
4. Sistem prikazuje formu za unos rasporeda ispita (IA)
5. Zaposleni u službi unosi podatke o rasporedu ispita (APUSO)
6. Zaposleni u službi saopštava sistemu da želi zapamti podatke o rasporedu ispita. (APSO)
7. Sistem pamti podatke o rasporedu ispita (SO)
8. Sistem saopštava korisniku da je uspešno sačuvao podatke o rasporedu ispita.(IA)

***Alternativni scenario:***

8.1 Sistem saopštava zaposlenom u službi o da snimanje podataka o rasporedu ispita nije uspelo. (IA)



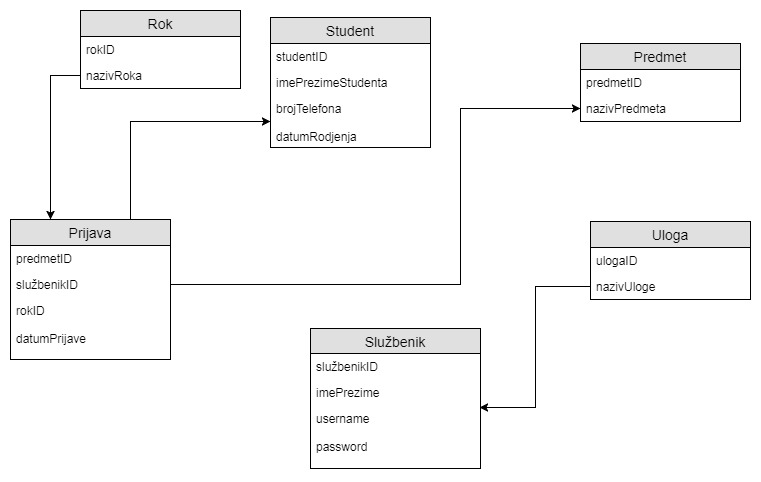
Slika 8: Dijagram klasa - raspored ispita



Slika 9: Dijagram sekvence - raspored ispita

# Arhitektura

## Dijagram klasa



U radu je korišćena baza podataka : služba.

Baza služba se sastoji od šest tabela: predmet, prijava,rok, službenik,student i uloga. U tabeli predmet se nalaze dve kolone: predmetID, nazivPredmeta. Tabela prijava se sastoji od 6 kolona : predmetID, brojIndeksa, rokID, sluzbenikID,ocena i datumPrijave. Tabela prijava se sastoji od 3 kolone : rokID, nazivRoka, školskaGodina. Tabela službenik se sastoji od 5 kolona : sluzbenikID, imePrezime, username, password i uloga. Tabela student se sastoji od 4 tabele : brojIndeksa, imePrezimeStudenta, brojTelefona i datumRodjenja. Tabela uloga se sastoji od 2 kolone : ulogaID i nazivUloge.

# Specifikacija REST API-ja

|  |  |
| --- | --- |
| Opis funkcije | Unesi predmet |
| HTTP metoda | POST |
| URL | /unesiPredmet |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute predmetID[int],nazivPredmeta[varchar]  Primer:  [  {  "predmetID":"1",  "nazivPredmeta":"ITEH",  },  {  "predmetID":"2",  "nazivPredmeta":"Projektovanje softvera",  },  {  "predmetID":"3",  "nazivPredmeta":"Multimediji",  }    ] |
| Format izlaznih parametara | application/json |

# Opis korišćenih tehnologija

U našem radu koristili smo sledeće tehnologije:

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

## HTML

HTML (engl. HyperText Markup Language, jezik za označavanje hiperteksta) je opisni jezik specijalno namenjen opisu 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.Standard održava Konzorcijum za Veb (W3C, World Wide Web Consortium).

## CSS

CSS je jezik osmišljen da ulepša web stranice. Stranice napisane samo HTML sintaksom nisu vizuelno atraktivne. CSS fajlovi su, po strukturi, takodje obični tekstualni fajlovi koji slede jednostavnu sintaksu pisanja CSS koda. CSS kod se može nalaziti unutar HTML dokumenta ali i kao samostalan dokumentsa ekstenzijom .css . CSS sintaksa se sastoji od opisa izgleda elemenata u dokumentu. Opis može da definiše izgled više elemenata, i više opisa može da definiše jedan element. Na taj način se opisi slažu jedan preko drugog da bi definisali konačni izgled određenog elementa (otuda naziv Cascading (engl. cascade - crijep) da bi se dočaralo slaganje jednog stila preko drugog u definisanju konačnog izgleda elementa)

Svaki opis se sastoji od tri elementa:

* definicija ciljnih elemenata
* svojstva
* vrednosti

## PHP

PHP (PHP: Hypertext Preprocessor) je specijalizovani skriptni programski jezik koji se koristi za izradu dinamičkih web stranica, tj. za dinamičko generisanje HTML koda.

Pomoću PHP-a možete kreirati HTML stranicu na serveru pre njenog slanja na lokalni računar klijentu popunjenu dinamičkim sadržajem. Na ovaj način PHP kod koji je generisao stranu se ne može videti, i prikazuje se čisti HTML kod.

Po sintaksi sličan je programskom jeziku C, čak ima i iste funkcije. To znači da jednu radnju možete izvesti korišćenjem više različitih funkcija.

PHP je nastao 1995. godine napravio Rasmus Lerdorf iz PHP/FI. Koristio je Perl skripte na svojim web stranicama. On je taj program nazvao “Alat za ličnu prezentaciju”. Kasnije je na to dodavao funkcije iz programskog jezika C za komunikaciju s bazama podataka i postavljanja na server dinamičkih web stranica.

PHP 3.0 je predstavljao prvu verziju koja dosta liči na današnje verzije PHP-a. Kreirali su ga izraelski programeri Endi Gutmans i Zeev Suraski 1997. godine, kao potpunu preradu projekta PHP/FI. Ime je promenjeno u PHP (bez dodatka 'FI'), čije se značenje izmenilo da predstavlja rekurzivni akronim 'PHP: preprocesor hiperteksta' (PHP: Hypertext Preprocessor ). PHP 4.0 je nastao 1998. godine, a verzija PHP 5.0 2004. godine koja je donela mnogo imena u Objektno Orijentisanom Programiranju.

Svaka PHP datoteka sadrži kvačice na početku i na kraju ‘<?php ?>’između kojih se nalazi php kod. Pored toga svaka promenljiva ima prefiks $. U nazivu promenljive nema razmaka. Kao razmak može se koristiti znak “\_”. Još jedna vrlo bitna stvar kod promenljivih u PHP-u je da su imena case-sensitive.

## AJAX(Asynchronous JavaScript And XML)

AJAX nije novi jezik programiranja, već tehnika za kreiranje boljih, bržih i interaktivnijih web aplikacija. Sa AJAX-om, vaš JavaScript može da komunicira direktno sa serverom, koristeći JavaScript XMLHttpRequest objekat. Pomoću ovog objekta, vaš JavaScript može da razmenjuje podatke sa web serverom, bez ponovnog učitavanja stranice.

AJAX koristi asinhroni transfer podataka (HTTP zahtevi) između pretraživača i web servera, mogućujući web stranicama da traže male bitove informacija od servera umesto čitavih stranica.AJAX tehnika programiranja omogućuje da Internet aplikacije budu manje, brže i prijateljskije za korisnika.To je tehnologija pretraživača koja je nezavisna od softvera web servera.

AJAX je zasnovan na sledećim web standardima:

* JavaScript
* XML
* HTML
* CSS

Web standardi koji se koriste u AJAX-u su dobro definisani i podržani od svih velikih web pretraživača. AJAX aplikacije su nezavisne od platforme i od pretraživača. Web aplikacije imaju mnogo prednosti nad desktop aplikacijama; one mogu da dopru do više korisnika, lakše su za instaliranje i obezbeđivanje podrške, i lakše za razvijanje.

Međutim, Internet aplikacije nisu uvek tako "bogate" i prijateljske prema korisnicima kao tradicionalne desktop aplikacije.Sa AJAX-om, Internet aplikacije mogu da budu bogatije i intuitivnije za korišćenje nego što su bile do sada.Nema ničeg novog što treba da naučite. AJAX je zasnovan na postojećim standardima. Ove standarde već nekoliko godina koriste programeri.

U tradicionalnom JavaScript kodiranju, ako želite da dobijete bilo kakvu informaciju iz baze podataka ili fajla koji se nalazi na serveru, ili pošaljete serveru informacije o korisniku, vi ćete trebati da napravite HTML formu i GET (dobijete) ili POST (pošaljete) podatke serveru. Korisnik treba da klikne na "Submit" dugme da bi poslao/dobio informaciju, sačeka odgovor servera, i onda će se učitati nova stranica sa rezultatima.

Pošto server vraća novu stranicu svaki put kada korisnik pošalje neke podatke, tradicionalne web aplikacije znaju da budu spore i manje prijateljske prema korisnicima.

Sa AJAX-om, vaš JavaScript komunicira direktno sa serverom, kroz JavaScript XMLHttpRequest objekat. Pomoću HTTP zahteva, web stranica može da pošalje zahtev serveru, i dobije odgovor od njega - bez ponovnog učitavanja web stranice. Korisnik ostaje na istoj stranici, i neće primetiti da je skript poslao zahtev za stranicom, ili da je u pozadini poslao podatke na server.

Koristeći XMLHttpRequest objekat, web programer može da ažurira stranicu sa podacima sa servera nakon što se stranica učitala!Kao što smo već spomenuli, AJAX je postao popularan 2005. godine zahvaljujući Google-u, tojest Google Suggest-u.

Google Suggest koristi XMLHttpRequest objekat kako bi kreirao veoma dinamički web interfejs: Kada počnete da kucate u Google-ovo polje za pretraživanje, JavaScript šalje slova serveru a server vraća listu predloga.

## JSON

JSON, odnosno JavaScript Object Notation, je tekstualno baziran otvoreni standard dizajniran za razumljivu ljudima razmenu podataka. Ona je izvedena iz Javaskript jezika za predstavljanje jednostavnih struktura podataka i asocijativni niz, nazvan objektima. Uprkos vezi sa JavaSkriptom, to je jezicki nezavistan, za mnoge jezike.

U okviru JSON-ove strukture mogu da se koriste sledeći tipovi podataka:

* Broj (JavaScript format u pokretnom zarezu sa dvostrukom preciznošću, zavisi od implementacije)
* String (Unicode format, sa dvostrukim navodnicima, kao izlazna sekvenca se koristi backslash )
* Boolean (true ili false)
* Niz (uredjena sekvenca vrednosti, odvojena zarezima i uokvirena kockastim zagradama; vrednosti ne moraju biti istog tipa)
* Objekat (neuredjena kolekcija ključ:vrednost parova sa ‘:’ karakterom koji razdvaja ključ i vrednost, razdvojeni zarezima i uokvireni vitičastim zagradama. Ključevi moraju biti niske i različiti od ostalih ključeva)
* null (prazno)
* Beline se mogu slobodno dodati izmedju strukturalnih karaktera (zagrada “{ } [ ]”, dve tačke “:” i zareza “,”).

Struktura može biti organizovana u vidu:

* Zbirke parova (ime / vrednost)

Na raznim jezicima, to je realizovano kao objekat, zapis, struktura, rečnik, heš tabela, lista sa ključevima ili asocijativni niz. Ova struktura se obeleažva sa { } vitičastim zagradama, a podaci u obliku ime/vrednost su odvojeni sa zarezom.

* Niza (uređena lista vrednosti)

Niz se obeležava sa [ ] uglastim zagradama, a članovi niza u odvojeni zarezom.

## 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.

## Flight

Flight je specijalizovani PHP framework namenjen prvenstveno razvoju REST servisa. Osnovna funkcionalnost ovog framework-a je rutiranje, odnosno jednostavno kreiranje funkcija koje odgovaraju na HTTP zahteve upućene korišćenjem neke standardne HTTP metode (GET, POST, PUT, DELETE). Flight je objektno-orijentisan framework i može se koristiti bilo pozivanjem statičkih metoda koje su definisane u klasama Flight-a, bilo instanciranjem novog objekta. U ovoj skripti će biti korišćene statičke metode. Moguće je definisanje sopstvenih globalnih promenljivih u Flight-u, segmentno rutiranje, korišćenje Flight-a za automatizaciju dodavanja HTML templejta i dr.

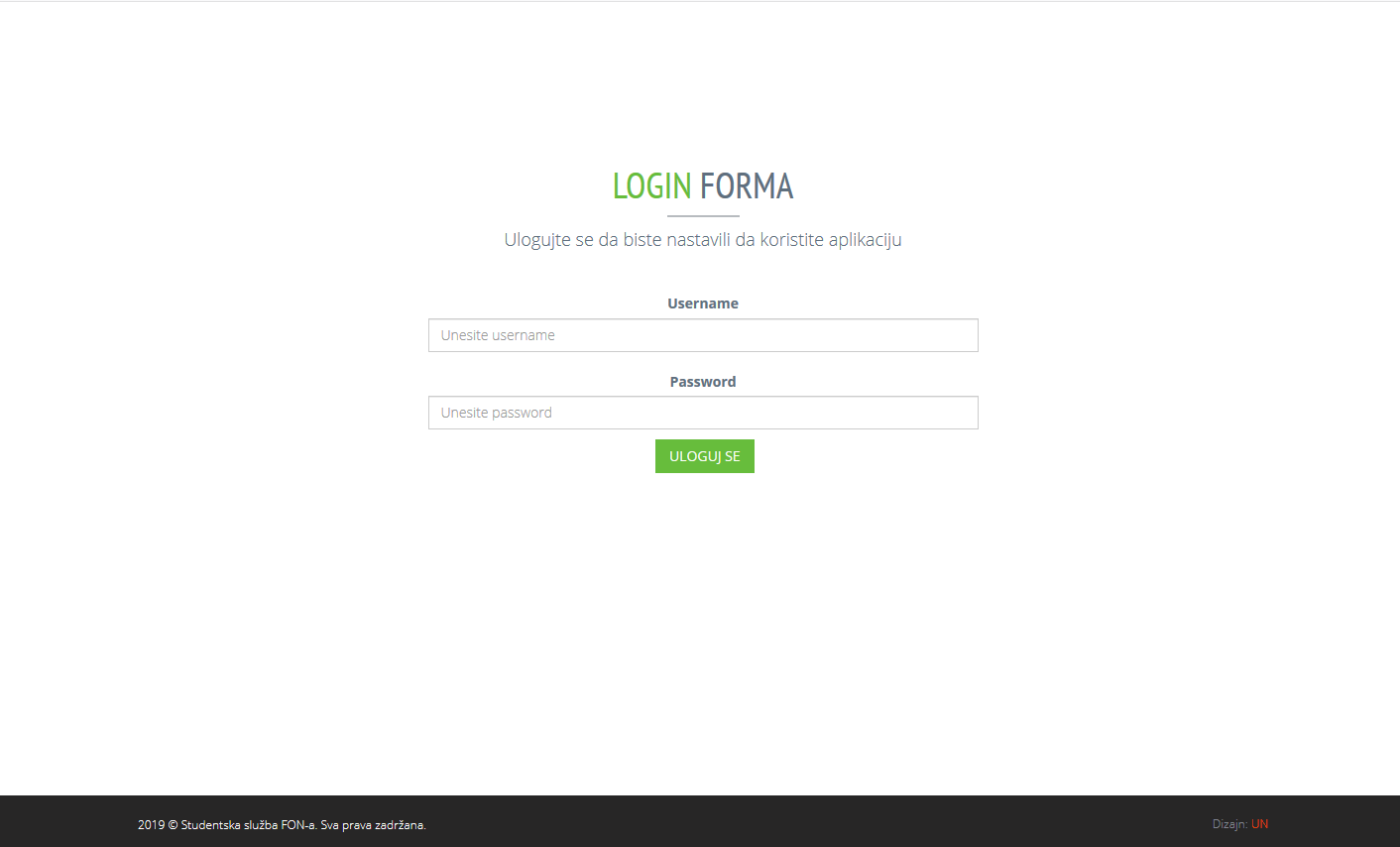
# Korisničko uputstvo

Aplikacija služi za rad u studentskoj službi fakulteta.

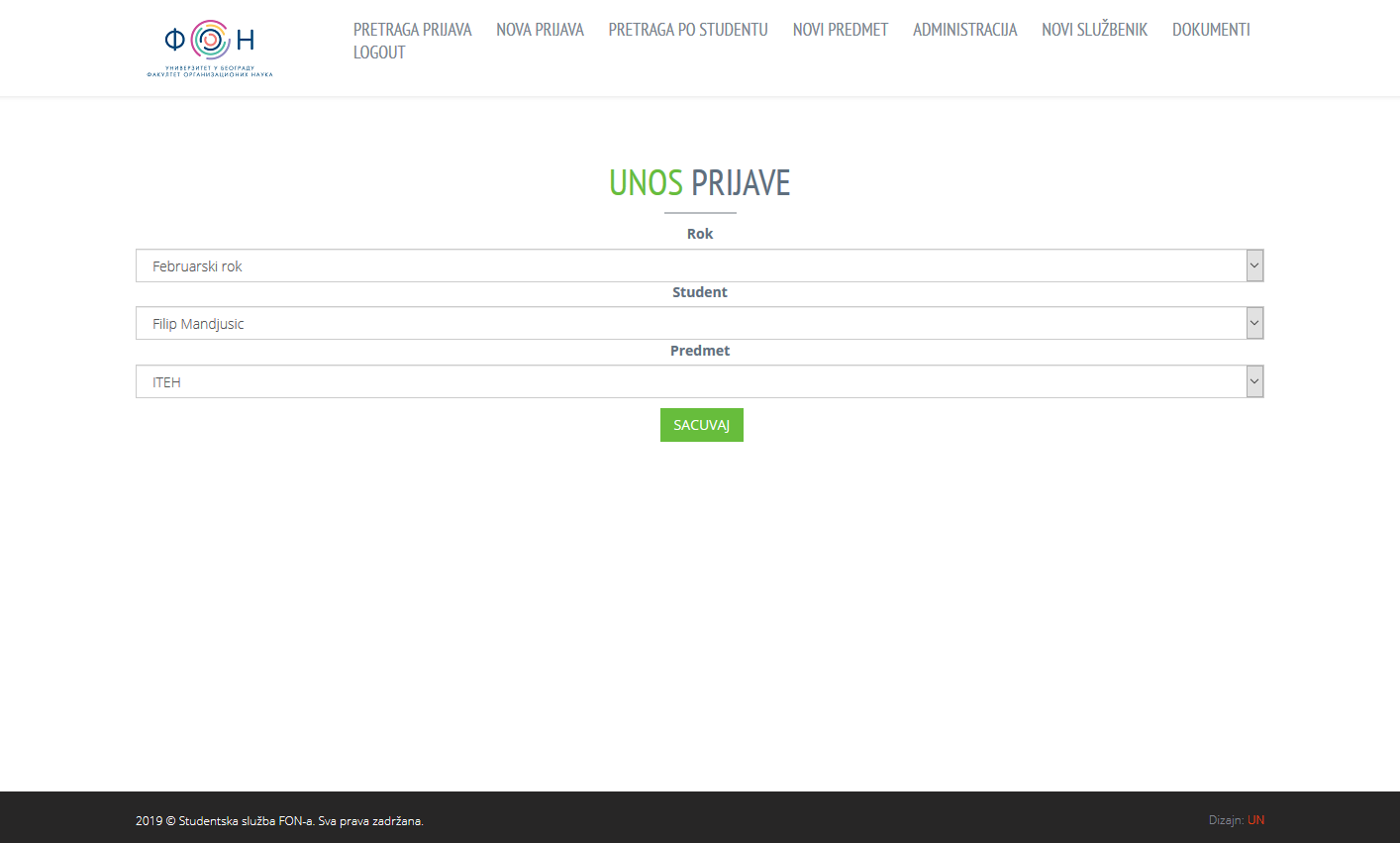


## Službenik

Da bi pristupio sistemu službenik mora da se uloguje.



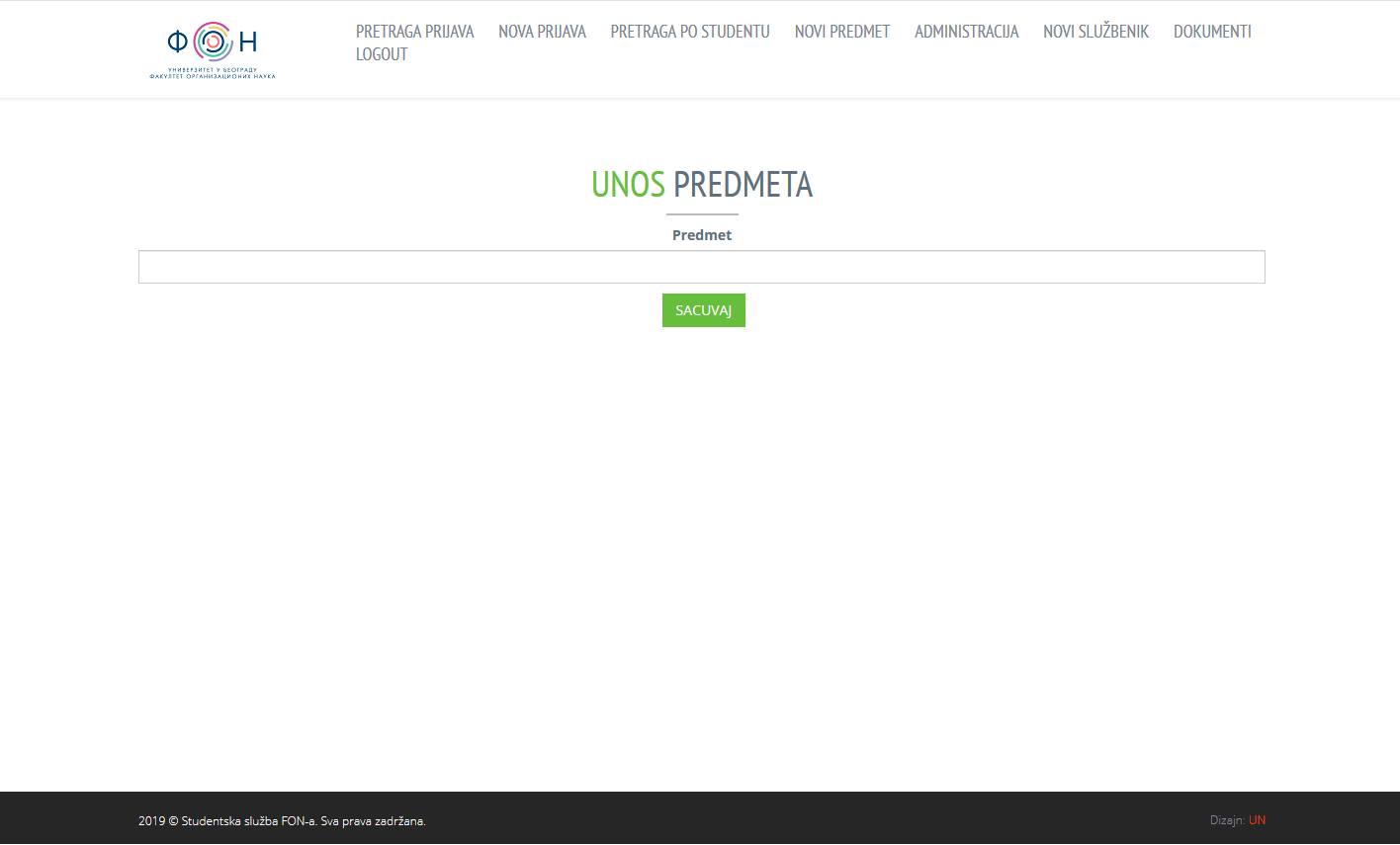
Nakon što se uloguje na početnoj strani po default-u se pojavljuje pretraga po rokovima, sa prikazom svih rokova. Nudi se opcija pretrage po rokovima iz liste. Sledeća stavka jeste nova prijava kojom se omogućuje prijava ispita u određenom roku za datog studenta.



Zatim je omogućena i pretraga po studentima koja nam daje sve relevantne informacije o zadatom studentu.



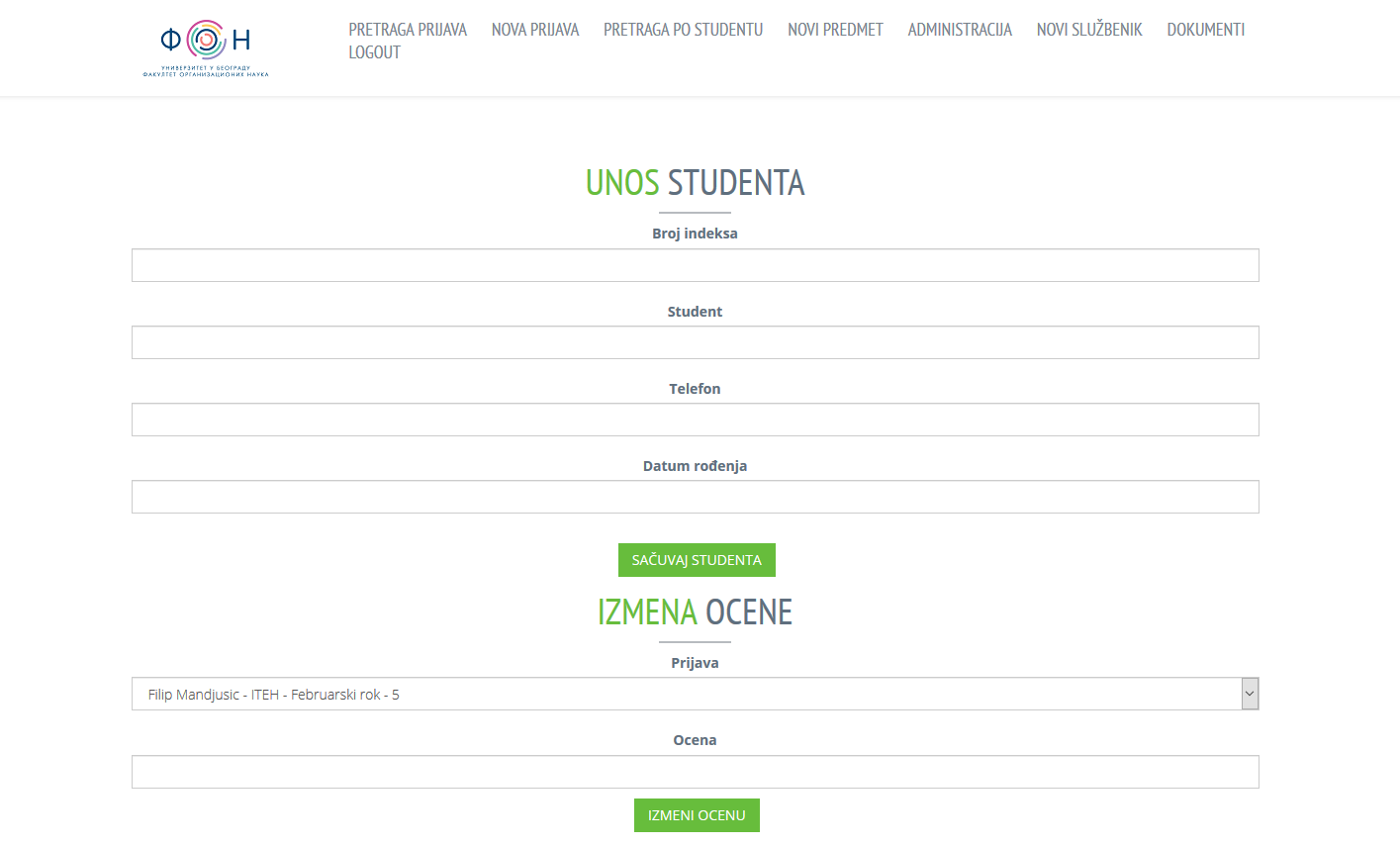
Kao opcija se javlja i unos novog predmeta.Nakon unosa predmeta prilikom sledećeg unosa nove prijave dodaje se i novi predmet kao jedna od opcija izbora.

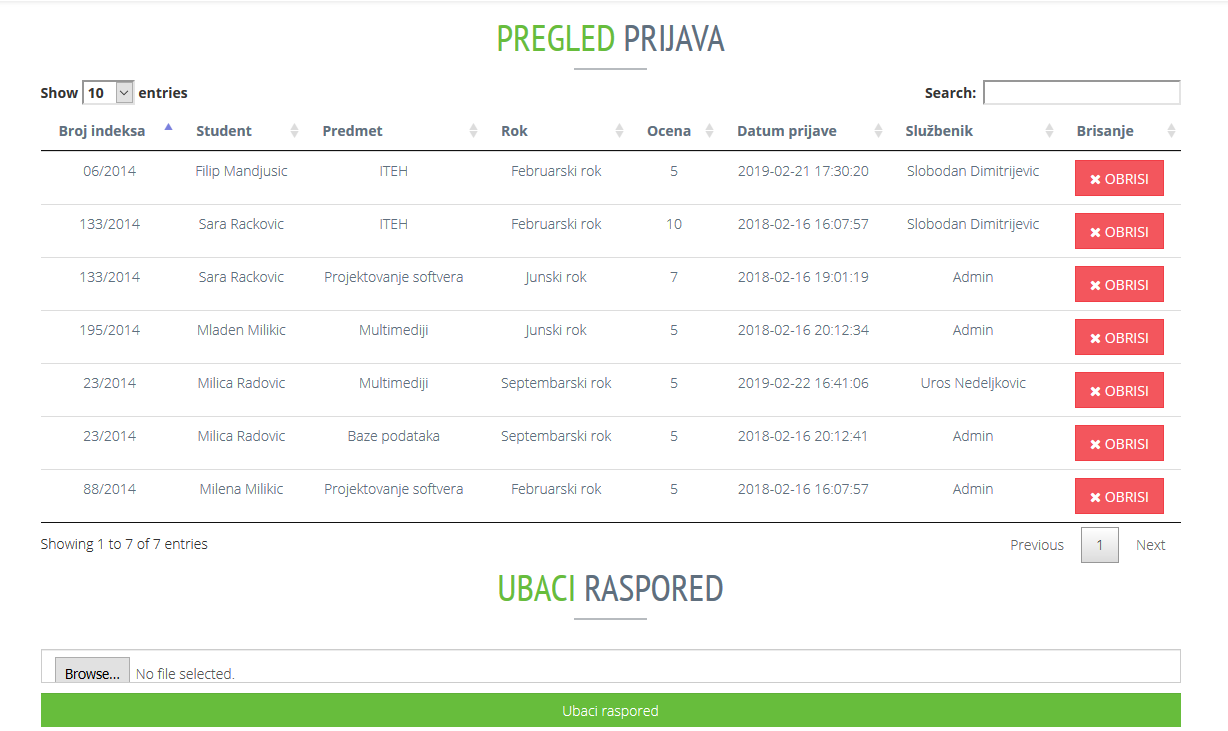


## Administrator

Ukoliko je službenik ulogovan kao administrator,njemu su omogućene dodatne opcije.

Poput unosa podataka o novom studentu u sistem. Zatim, mogućnost izmene ocena,kao i pregled prijava gde je omogućena pretraga,kao i brisanje istih.





Administrator takođe može i da upload-uje raspored ispita,kojem se posle pristupa u okviru fajlova. Vizuelni podaci pružaju grafički prikaz broja prijava po predmetu na pie dijagramu. Pored toga, administrator može da kreira novog službenika koji će imati pristup aplikaciji.

