

Automatizacija sistema studentskog doma prilikom konkurisanja i useljavanja

Novica Lazić
Fakultet tehničkih nauka
Univerzitet u Novom Sadu
Trg Dositeja Obradovića 6, 21000 Novi Sad
lazic.sr.9.2020@uns.ac.rs

SAŽETAK

Rad istražuje potrebu za automatizacijom sistema studentskog doma kako bi se brže i efikasnije sproveo proces useljavanja u dom. Uz pomoć naše aplikacije je omogućeno lakše sprovođenje konkursa i lakše snalaženje studenata bez nepotrebnih gužvi u službama.

studentski dom; automatizacija; organizacija;

1. UVOD

Praktično upravljanje studentskim domom je ključno kako bi se omogućila bolja organizacija podataka o studentima i smeštaju. Ovaj rad omogućava lakšu prijavu za studente bez silne papirologije i gubljenja vremena kao i lakše snalaženje prilikom popunjavanja podataka i lakšeg pronalaženja potrebnih informacija.

Da bismo olakšali prijavu i obaveštavanje studenata o konkursu za smeštaj jako je bitno da sajt bude jasan i pregledan, gde smo primer loše prakse imali sajt scns[1] (neaužrnost i nesnalažljivost prilikom potrebnih informacija).

2. PREGLED KORIŠĆENIH TEHNOLOGIJA I ALATA

Za potrebe projekta iz Euprave u servisu 'Dom' koristimo Spring Boot[3] za backend, Angular[2] za klijentski dio aplikacije i SQL[4] kao bazu za skladištenje svih podataka u projektu.

Krenućemo od serverske strane odnosno backend-a za koji koristimo Spring Boot koji nam uz pomoć svog alata Mavena omogućava lakše upravljanje projektom. Kroz Maven smo definisali poslovnu logiku, zavisnosti i konfiguraciju našeg sajta.

SQL koristimo za struktuiramo skladištenje podataka, a i za upravljanje i pribavljanje neophodnih podataka iz baze. Za mapiranje objekata u bazu koristimo JPA koji nam omogućava rukovanje podacima iz Java aplikacije.

Za frontend deo smo koristili Angular i njegov alat Angular CLI uz pomoć kog smo generisali klijentski dio našeg sajta.

Zahvaljujući RESTful servisima povezali smo efikasno prenos podataka između frontend i backend dijela našega sajta.

Kroz ovu kombinaciju tehnologija i alata, ostvarili smo integraciju između backend i frontend komponenti, pružajući korisnicima funkcionalan i efikasan sistem za upravljanje podacima u okviru projekta "Dom".

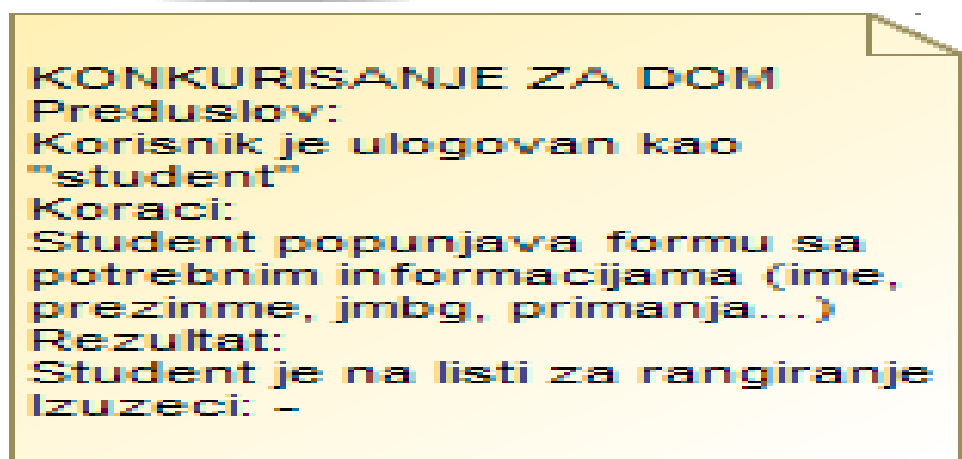
3. SPECIFIKACIJA ZAHTJEVA

U ovom poglavlju se objašnjeni funkcionalni i nefunkcionalni zahtjevi softverskog rešenja predstavljenog u ovom radu.

3.1 SPECIFIKACIJA FUNKCIONALNIH ZAHTJEVA

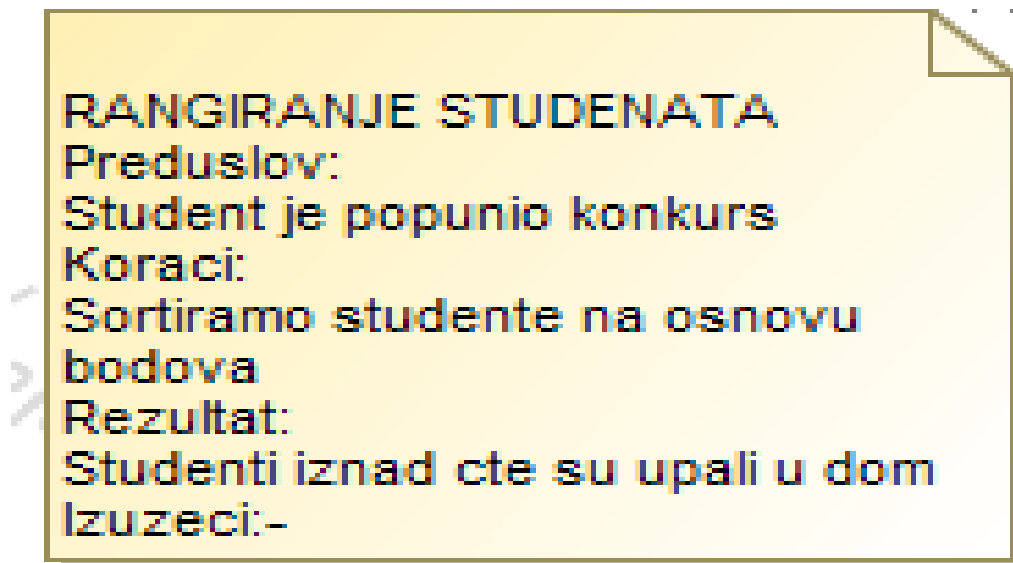
U ovom odeljku, opisujemo funkcionalne zahtjeve koristeći UML dijagrame slučajeva korišćenja.

Slika 1 prikazuje opis slučaja 'Konkurisanja za dom'.



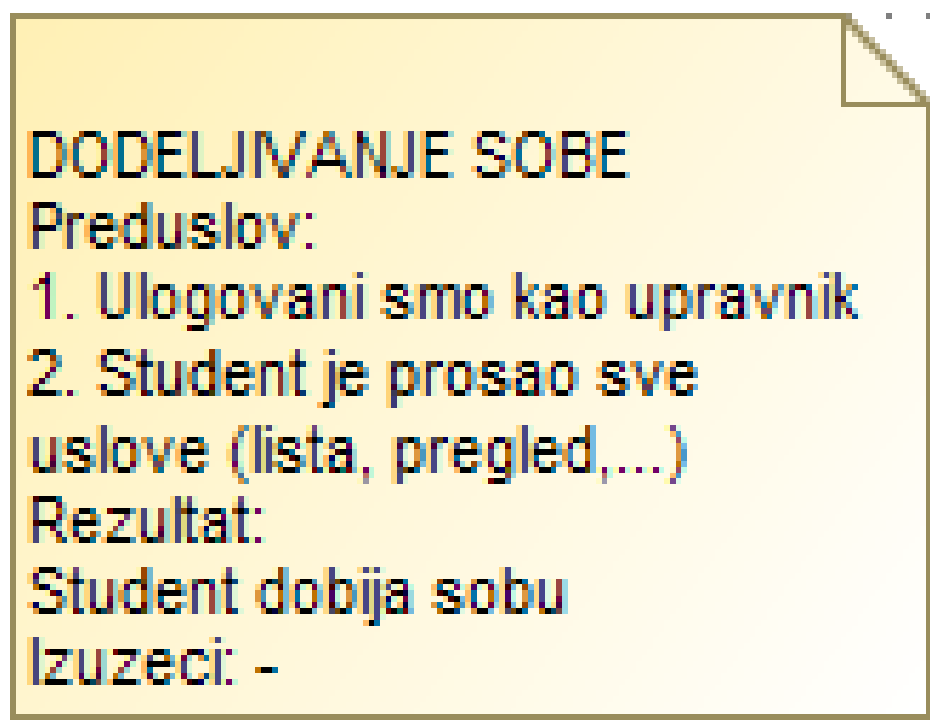
Slika 1 – Konkurisanje za dom

Slika 2 prikazuje opis slučaja 'Rangiranja studenata za dom'.



Slika 2 – Rangiranje studenata

Slika 3 prikazuje opis slučaja 'Dodjeljivanja sobe studentu'.



Slika 3 – Dodjeljivanje sobe

3.2 SPECIFIKACIJA NEFUNKCIONALNIH ZAHTEJEVA

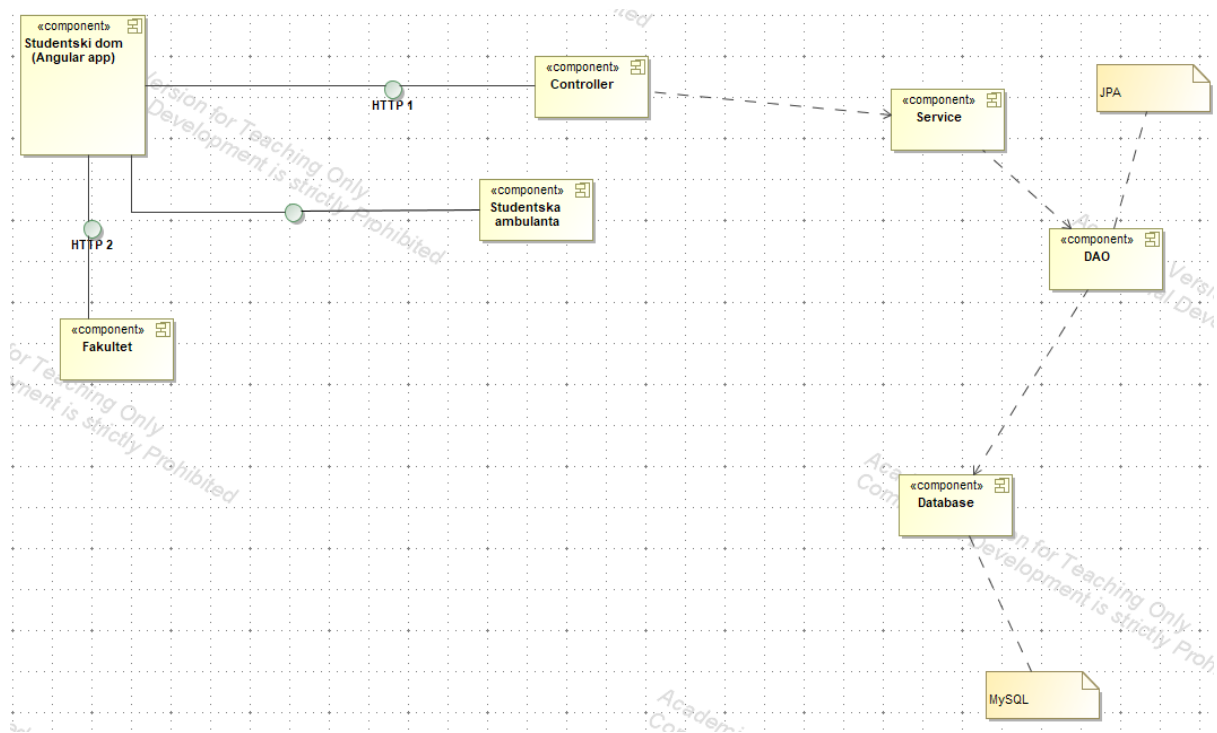
Specifikacija nefunkcionalnih zahtjeva definiše svojstva sistema koja nisu vezana za konkretne funkcionalnosti, već se odnose na performanse, sigurnost, upotrebljivost i druge aspekte. Neki od nefunkcionalnih zahteva uključuju:

Performanse: Sistem je namenjen za izvršavanje na desktop računarima (Windows, macOS, Linux).

Bezbednost: Podaci korisnika su zaštićeni tako što prilikom registracije na naš sajt lozinka se čuva tako što koristimo hash&salt za čuvanje lozinke, a takođe smo i onemogućili pristup svim korisnicima sajta koji nisu ulogovani.

4. SPECIFIKACIJA DIZAJNA (ARHITEKTURA SISTEMA)

Ovo poglavlje objašnjava dizajn softverskog rešenja za sistem studentskog doma. Detaljno ćemo opisati arhitekturu sistema, komponente koje ga čine i način njihove međusobne komunikacije na slici broj 5.



SLIKA 5. ARHITEKTURA SISTEMA

Na slici 5. komponente Studentski dom, Fakultet i Studentska ambulanta predstavljaju korisnički interfejs web sajta.

Controller-i (slika 5.) upravljaju komunikacijom između serverskog i klijenskog dijela projekta.

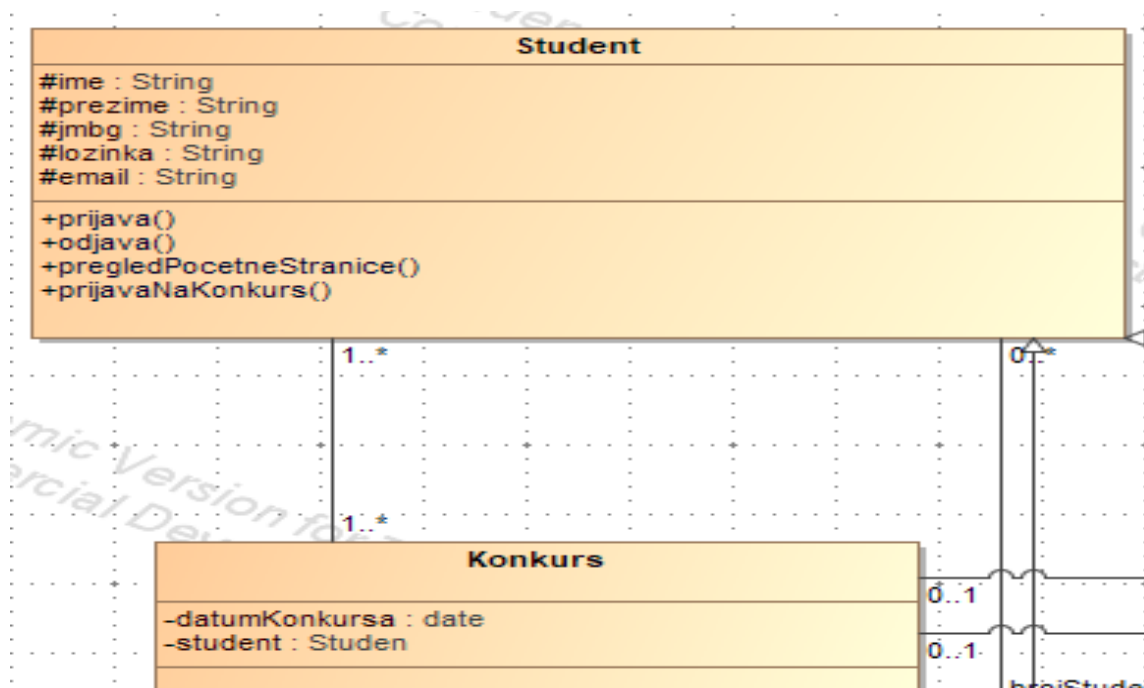
Service-i (slika 5.) predstavljaju poslovnu logiku sistema.

DTO-ovi (slika 5.) se koriste za prenos podataka između slojeva sistema, a u specifičnom ovom sistemu koristimo JPA zajedno sa DTO za vezu između baze podataka i Java objekata.

Za komunikaciju između komponenata (controller, service, korisnički interfejs) koristimo RESTful API, gdje za svaku komponentu korisničkog interfejsa koristimo različite endpoint-e.

5. SPECIFIKACIJA DIZAJNA (CLASS DIAGRAM)

Na slici 6 je pomoću UML dijagrama klasa predstavljen objektni model sistema.



Slika 6. Class diagram

Klasa Student reprezentuje studenta i ima njegove lične podatke i podatke za autentifikaciju na sistem.

Konkurs sadrži podatke o studentu koji su se prijavili na određeni konkurs.

Kardinalitet između ove dvije klase prikazuje da se jedan ili više studenata može prijaviti na konkurs, a i da se jedan student može prijaviti na više konkursa.

6. IMPLEMENTACIJA

Ovo poglavlje detaljno opisuje implementaciju rešenja, fokusirajući se na zahtjeve definisane u specifikaciji zahtjeva i dizajnu sistema.

6.1 IMPLEMENTACIJA SERVERSKOG DIJELA APLIKACIJE

U prvom delu ovog poglavlja, biće objašnjena implementacija klijentskog dela aplikacije. Ključni dijelovi koda biće prikazani kroz listinge sa odgovarajućim objašnjenjima.

Za prijavu studenta na konkurs je implementirana metoda prikazana na listingu 1.

```
public void prijavistiudentaNaKonkurs(StudentDTO studentDTO) {  
    String username = studentDTO.getUsername();  
  
    Student student = studentRepository.findByUsername(username)  
        .orElseThrow(() -> new IllegalArgumentException("Student sa datim korisničkim imenom ne postoji"));  
  
    student.setGodinaStudiranja(studentDTO.getGodinaStudiranja());  
    student.setOsvojeniBodovi(studentDTO.getOsvojeniBodovi());  
    student.setProsek(studentDTO.getProsek());  
  
    double bodovi = studentDTO.getGodinaStudiranja() + (double) studentDTO.getOsvojeniBodovi() / studentDTO.getProsek();  
    student.setBodovi(bodovi);  
  
    Konkurs konkurs = konkursService.findKonkursById(studentDTO.getKonkursId())  
        .orElseThrow(() -> new IllegalArgumentException("Konkurs sa datim ID-om ne postoji"));  
    student.setKonkurs(konkurs);  
  
    studentRepository.save(student);  
}
```

LISTING 1 – PRIJAVA STUDENTA NA KONKURS

Ovoj metodi se prosleđuje referenca na objekat koji reprezentuje studenta.

Iz objekta studentDTO dobijamo korisničko ime studenta i pronalazimo ga u bazi, kad ga nađemo unosimo podatke za godinu studiranja, osvojene bodove i prosek, bodove računamo po formuli bodovi, koristeći konkurs servis pronalazimo konkurs sa prosleđenim ID-jem i na kraju ažuriramo studenta u bazi podataka.

Za prikaz konačne rang liste studenata je implementirana metoda prikazana na listingu 2.

```
@GetMapping("/rang-lista-soba")
public ResponseEntity<List<Student>> getRangListaSobaNull() {
    List<Student> rangLista = studentRepository.findByBodoviGreaterThanAndSobaIsNull(0);

    if (rangLista != null && !rangLista.isEmpty()) {
        return ResponseEntity.ok(rangLista);
    } else {
        return ResponseEntity.noContent().build();
    }
}
```

LISTING 2 – KONAČNA RANG LISTA

Koristimo studentRepository i da pronađemo sve studente kod kojih je kolona bodova veća od nula i soba jednako null i ako uspješno pronađemo studenta kreiramo listu sa svim studentima koji ispunjavaju uslov.

Za prikaz dodjele sobe studentu je implementirana metoda prikazana na listingu 3.

```
@PostMapping("/dodeli-sobu")
public ResponseEntity<Void> dodeliSobu(@RequestParam String username, @RequestParam Long sobaId) {
    try {
        Optional<Student> studentOptional = studentRepository.findByUsername(username);
        Optional<Soba> sobaOptional = sobaRepository.findById(sobaId);

        if (studentOptional.isPresent() && sobaOptional.isPresent()) {
            Student student = studentOptional.get();
            Soba soba = sobaOptional.get();

            student.setSoba(soba);

            studentRepository.save(student);

            return ResponseEntity.ok().build();
        } else {
            return ResponseEntity.notFound().build();
        }
    } catch (Exception e) {
        return ResponseEntity.badRequest().build();
    }
}
```

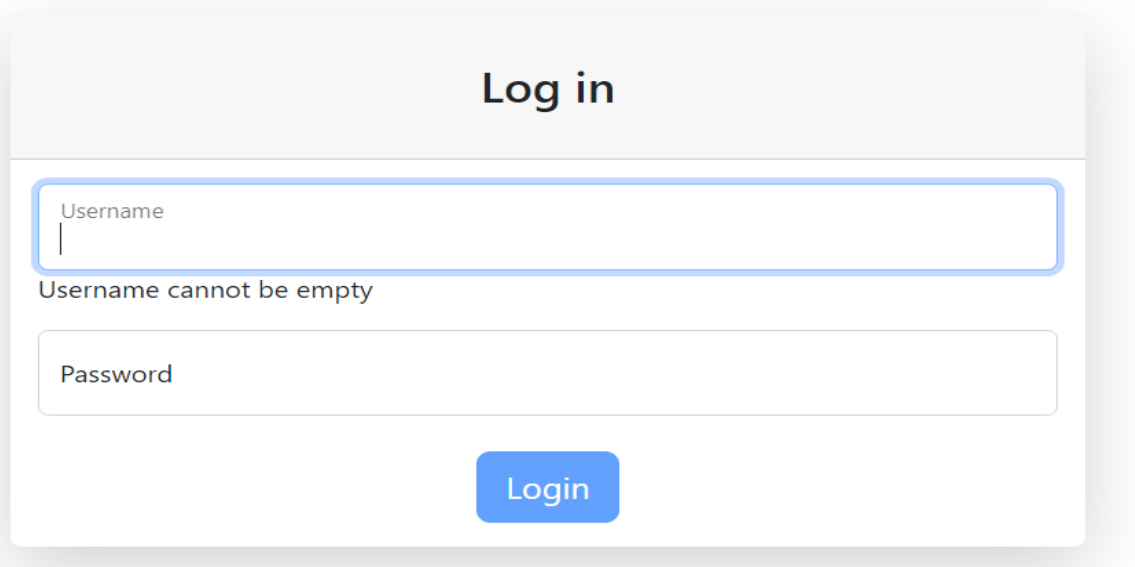
LISTING 3 – DODJELA SOBE

Koristimo studentRepository i sobaRepository da bi pronašli studenta na osnovu username-a i sobu na osnovu sobald i kada ispunimo uslove ažuriramo studenta i kolonu sobald u tabeli student.

7. DEMONSTRACIJA

Ovo poglavlje prikazuje korake korišćenja web sajta studentskog doma. Demonstracija će pratiti tipičan tok korišćenja sajta pružajući jasne i korisne informacije o ključnim funkcionalnostima sistema.

U centralnom dijelu stranice (slika 7) je forma za prijavu na sajt gdje unosimo username i password i ukoliko su podaci ispravni preusmeriće nas na početnu stranicu za studenta (slika 8) ili za upravnika (slika 9) u zavisnosti od role.



Log in

Username

Username cannot be empty

Password

Login

SLIKA 7 – LOGIN FORMA

Dostupni Konkursi

- **Beograd**

2024/2025

- **Novi Sad**

2024/2025

Prijavi se na konkurs

Nazad

Rang Lista

Prikazi informacije o sobi

SLIKA 8 – POČETNA STUDENT

Konkursi

- **Beograd**

2024/2025

- **Novi Sad**

2024/2025

Nazad

Konacna rang Lista

SLIKA 9 – POČETNA UPRAVNIK

7.1 FRONT DIO ZA STUDENTA

Na početnoj stranici za studenta (slika 8) imamo prikazane sve konkurse koji su još aktivni, a tu su i dugmad za prijavu na konkurs, rang lista i informacije o dobijenoj sobi.

Prilikom prijave na konkurs (slika 10) popunjavamo formu za prijavu sa traženim podacima i ukoliko se uspješno prijavimo vraća nas na početni ekran.

Prijava na konkurs

Korisničko ime:

Godina studiranja:

Osvojeni bodovi:

Prosek:

ID konkursa:

SLIKA 10 – FORMA ZA PRIJAVU

Nakon toga možemo i pogledati rang listu gdje su svi prijavljeni studenti.

Nakon što se završi konkurs i upravnik nam dodijeli sobu sve informacije možemo vidjeti na stranici o sobi (slika 11).

Prikaz informacija o sobi

Korisničko ime:

Soba ID: 1

Ime: Marko

Prezime: Marković

Username: maja123

SLIKA 11 – BROJ SOBE

7.2 FRONT DIO ZA UPRAVNIKA

Na početnoj stranici za upravnika (slika 9) imamo prikazane sve konkurse, a ispod njih imamo button da vidimo konačnu rang listu (slika 12).

Konacna rang Lista

- Marko Marković - Bodovi: 15

SLIKA 12 – KONAČNA RANG LISTA

Nakon toga imamo mogućnost za dodjelu sobe gdje nam se prikazuje forma za dodjelu (slika 13).

Dodeli sobu studentu

Korisničko ime: ID sobe:

SLIKA 13 – FORMA DODJELA SOBE

8. ZAKLJUČAK

Ovaj rad je istraživao potrebu za automatizacijom sistema studentskog doma kako bi se unapredio proces useljenja i olakšalo snalaženje studenata.

Dobra strana je da se sad sve može obaviti onlajn bez ogromnog gubljenja vremena, zaboravljenih papira... ali problem je za studente koji nemaju računar i ne mogu koristiti našu aplikaciju zato bi se mogla napraviti i mobilna aplikacija uz pomoć Swift-a ili Android studija.

9. LITERATURA

[1] <https://www.scns.rs/>

[2] <https://angular.io/>

[3] <https://spring.io/projects/spring-boot/>

[4] <https://www.mysql.com/>