

# Aplikacija za Hotel

**Profesor:** prof. dr Vladislav Miškovic

**Asistent:** Milan Tair

**Student:** Aleksa Cvetkovic 2016203643

## *Sadržaj*

Uvod	3
Cilj razvoja	3
Obim sistema	3

Prikaz proizvoda	3
Perspektiva proizvoda	3
Funkcije proizvoda	4
Karakteristike korisnika	4
Ograničenja	5
Definicije	5
Reference	5
Specifikacija zahteva	5
Spoljašnji interfejsi	5
Funkcije	6
Pogodnost za upotrebu	8
Zahtevane performanse	9
Zahtevi baze podataka	9
Projektna ograničenja	9
Sistemske karakteristike softvera	10
Dopunske informacije	10
Verifikacija	10
Spoljašnji interfejsi	10
Funkcije	10
Pogodnost za upotrebu	10
Zahtevane performanse	10
Zahtevi baze podataka	10
Projektna ograničenja	10
Sistemske karakteristike softvera	10
Dopunske informacije	10
Prilozi	10
Pretpostavke i zavisnosti	10
Akronimi i skraćenice	10

## *Uvod*

Prateća dokumentacija detaljno opisuje cilj razvoja same aplikacije, obim sistema, prikaz okruženja u kojem sistem funkcioniše, način na koji sistem funkcioniše, kao i pojedine nefunkcionalne osobine aplikacije. Takođe su prikazani određeni elementi same aplikacije, dijagrami, model baze, a nevedena su i određena ograničenja koja je neophodno ispoštovati.

## *Cilj razvoja*

Cilj razvoja aplikacije jeste mogućnost pouzdanog, brzog i jednostavnog načina pregleda i rezervacija hotelskih prostorija.

## *Obim sistema*

Sistem omogućava korisnicima da pregledaju hotelske prostorije po određenim filterima, i da ih rezervisu. Što se tiče administratora, imaju privilegije da dodaju nove prostorije, uređuju ih i prihvataju ili odbijaju rezervacije korisnika. Sami posetioци imaju mogućnost pregleda hotelskih prostorija, mogućnost filtriranja istih, kao i njihovo sortiranje po određenom parametru. Posetioци se mogu registrovati, dok se sami korisnici loguju svojim pristupnim parametrima kao administratori ili samo kao korisnici.

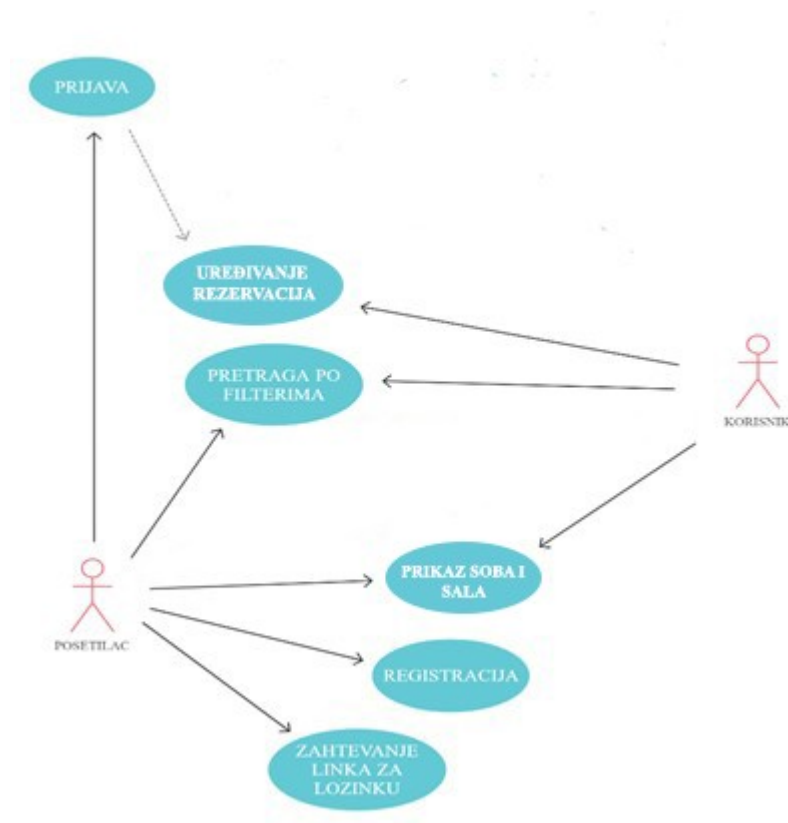
## *Prikaz proizvoda*

Prikaz proizvoda definiše perspektivu samog proizvoda i prikazuje funkcije posetioca i prijavljenog korisnika.

## *Perspektiva proizvoda*

Aplikacija omogućava jednostavnu rezervaciju hotelskih prostorija, bez mnogo napora i u jako kratkom vremenskom periodu. Aplikacija ne zahteva posebno iskustvo ni iz jedne oblasti niti neko prethodno znanje, tako da bi trebalo da bude laka za korišćenje svim korisnicima.

## Funkcije proizvoda



Funkcije koje obavljaju korisnici:

Posetilac se može registrovati, odnosno prijaviti na aplikaciju, može zahtevati link za lozinku, može pogledati same proizvode ( po kategorijama ili ovako ), kao što ih može i filtrirati u zavisnosti od interesovanja.

Prijavljeni korisnik ima iste privilegije kao i posetilac s tim da je on već prijavljen na aplikaciju, takođe može da pregleda proizvode kao i da ih pretražuje po kategorijama ili filterima. Prijavljeni korisnik, za razliku od posetioca, može rezervisati određene prostorije Hotela.

## Karakteristike korisnika

Aplikacija je namenjena za iskusne korisnike, ali i za korisnike koji nisu ranije koristili neku sličnu aplikaciju. Interfejs je dizajniran da bude lak za korišćenje i nije neophodno nikakvo prethodno iskustvo ili obrazovanje za njegovo korišćenje. Jedino što je korisniku potrebno jeste pristup Internetu.

## Ograničenja

Što se ograničenja tiče posetioci imaju najmanje mogućnosti korišćenja aplikacije – oni mogu samo pregledati prostorije po određenim parametrima, dok prijavljeni korisnici mogu i rezervisati prostorije. Administratori su jedini koji mogu menjati parametre prostorija, i jedini koji mogu dodavati i uklanjati određene prostorije

## Definicije

Sistem je primenjiv u svim slučajevima, a ne postoje posebne definicije koje bi dale značaja u primeni ove aplikacije.

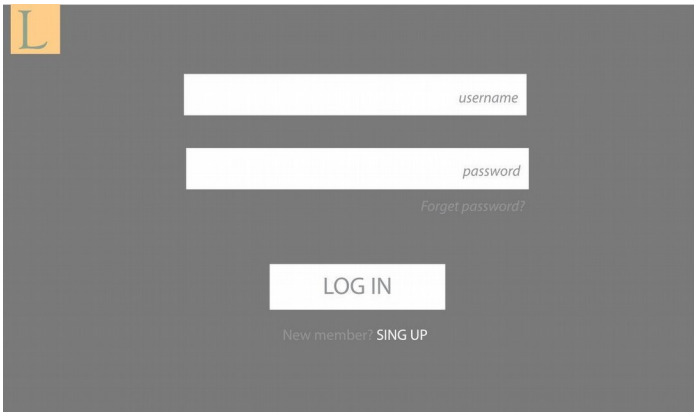
## Reference

“Aplikacija za Hotel”

<http://zadatak.singidunum.ac.rs/app/piivt-biranje-tema/?action=topic&id=59>

## Specifikacija zahteva

### Spoljašnji interfejsi



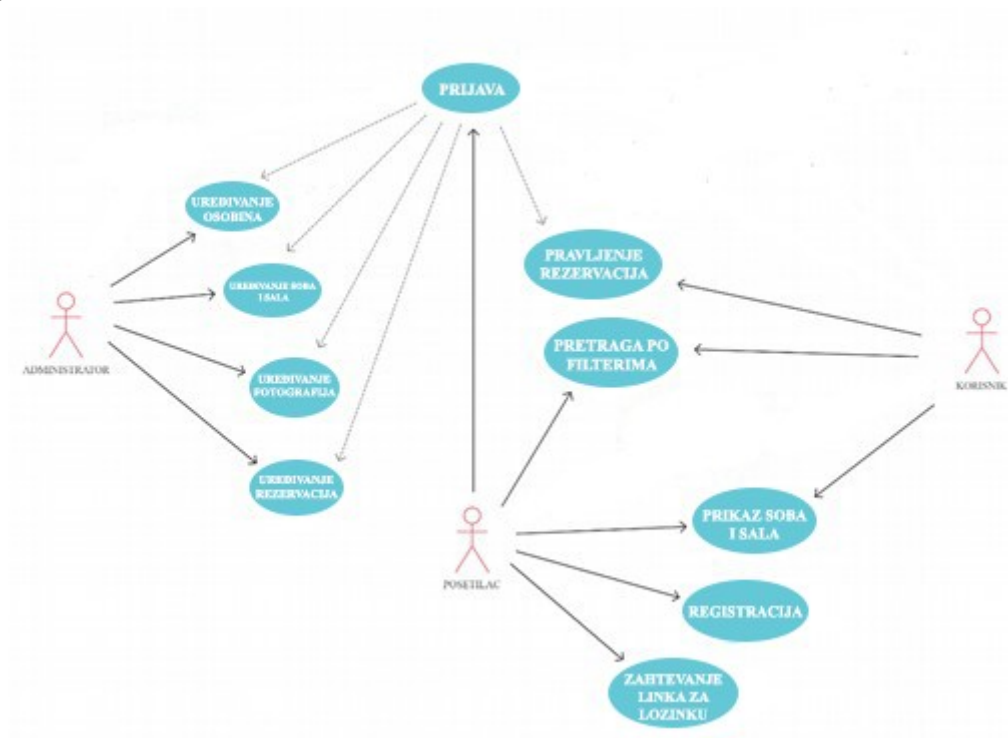
Stranica za prijavu, ili mogućnost registracije novog korisnika. Unosom parametara, korisnik se na sistem loguje kao običan korisnik ili kao administrator. Ukoliko zaboravi lozinku, ima mogućnost preuzimanja nove, a ukoliko nema nalog, ima mogućnost njegovor pravljenja.

Stranica za registraciju novog korisnika. Da bi napravio nalog, neophodno je da unese parametre koji su važni za dalje aktivnosti u samoj aplikaciji. Klikom na sign up dugme, posetilac postaje korisnik i može regularno rezervisati prostorije hotela.

. Svaka prostorija ima vidljivu količinu i cenu. Klikom na neku od prostorija otvara se stranica koja je detaljnije opisuje.

Stranica koja prikazuje poslednji korak u rezervaciji prostorije. , Sadrzi mogućnost za registraciju korisnika, kao i fotografiju sa detaljnim opisom prostorije, ukoliko korisnik želi rezervisati određenu prostoriju, prikazana je opcija za prijavu korisnika ukoliko nije prijavljen na sajt.

## Funkcije



Funkcije koje obavljaju:

Administrator ima mogućnost uređivanja kategorija, osobina, fotografija, proizvoda i samih porudžbina, kao njihovo dodavanje ili brisanje u zavisnosti od stanja sistema u datom trenutku.

Posetilac se može registrovati, odnosno prijaviti na aplikaciju, može zahtevati link za lozinku, može pogledati same proizvode ( po kategorijama ili ovako ), kao što ih može i filtrirati u zavisnosti od interesovanja.

Prijavljeni korisnik ima iste privilegije kao i posetilac s tim da je on već prijavljen na aplikaciju, takođe može da pregleda proizvode kao i da ih pretražuje po kategorijama ili filterima. Prijavljeni korisnik, za razliku od posetioca, može rezervisati prostorije, kao i promeniti ili ukloniti rezervaciju.

## Pogodnost za upotrebu

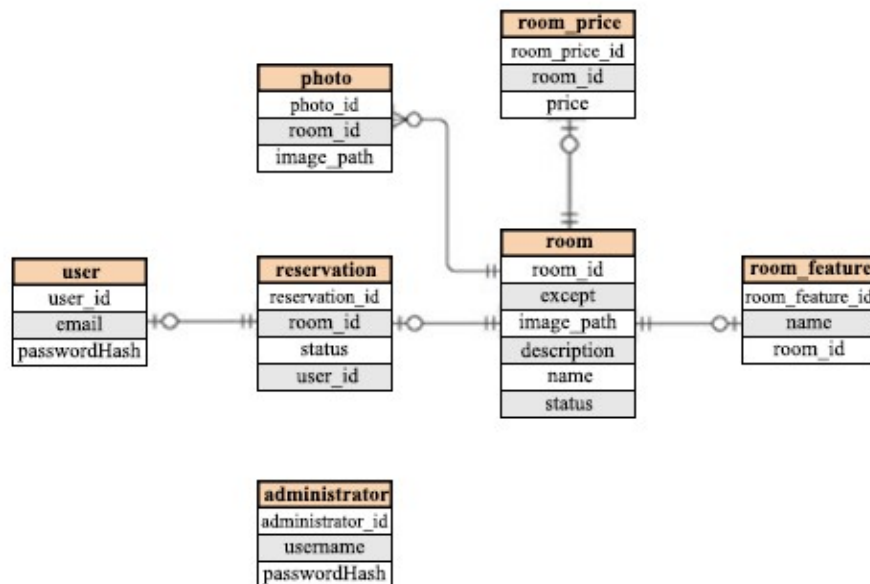
Aplikacija zadovoljava efektivnost i efikasnost korišćenja i predstavlja sistem koji pozitivno utiče na zadovoljstvo krajnjih korisnika.

## Zahtevane performanse

Vreme odziva Web aplikacije je minimalno, a broj istovremenih korisnika ne remeti rad same aplikacije i ne predstavlja problem samom sistemu niti usporeva njegov rad.

## Zahtevi baze podataka

Baza podataka mora da bude relacionala i treba koristiti MySQL/MariaDB RDBMS.



## Projektna ograničenja

Projekat treba biti realizovan na Node.js platform korišćenjem Nest.js razvojnog okvira i sam kod treba biti organizovan prema pravilima MVC arhitekture. Baza podataka mora da bude relacionala i treba koristiti MySQL/MariaDB RDBMS. Izrada projekta mora da bude sprovedena korišćenjem alata za verziranje koda Git.



## *Sistemske karakteristike softvera*

Posebna pažnja je obraćena na realizaciju zahteva pouzdanosti, responsive dizajna, bezbednosti sistema, pogodnosti za održavanje aplikacije, prenosivosti i raspoloživosti.

## *Dopunske informacije*

Za sada ne postoje nikakve dopunske informacije.

## *Verifikacija*

### *Spoljašnji interfejsi*

\*problem se javio kod implementacije tokena za administratora nakon čega je objavljena ista greska za funkcije koje administrator obavlja

\*problem je nastao takodje oko realizacije rezervacije korisnika, zbog čega je ona izostavljena.

## *Funkcije*

Administrator:

- Loguje se pristupnim parametrima

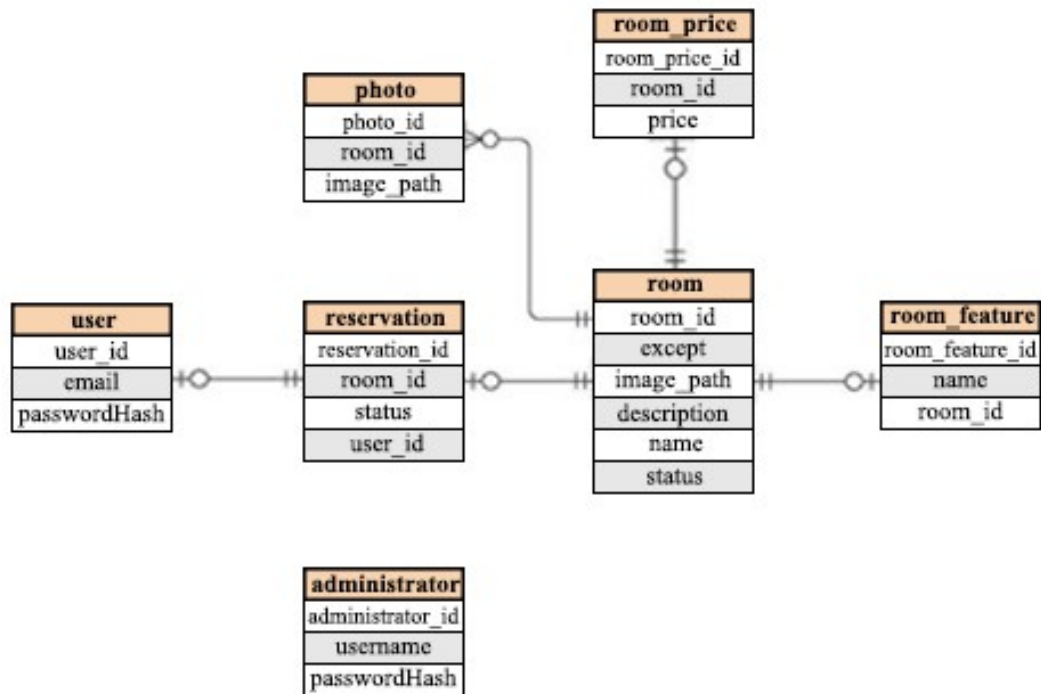
POST ▼	http://localhost:3000/auth/administrator/login	Send ▼
--------	--	--------

## *Pogodnost za upotrebu*

Aplikacija je laka za upotrebu, interfejs dizajniran da i korisnici bez prethodnog koriscenja mogu da je koriste bez problema.

*Zahtevane performanse*

*Zahtevi baze podataka*



Baza podataka ima sve potrebne relacije, kao i spoljasnje kljuceve definisane prema potrebi aplikacije i koriscenju tabela.

## Projektna ograničenja

Projekat jeste realizovan na Node.js platform korišćenjem Nest.js razvojnog okvira i sam kod je organizovan prema pravilima MVC arhitekture. Baza podataka je relaciona i koristi MySQL/MariaDB RDBMS. Izrada projekta sprovedena je korišćenjem alata za verziranje koda Git.

## Sistemske karakteristike softvera

registracija i prijavljivanje korisnika / dto :

```
@Post('user/register')
async userRegister(@Body() data: UserRegistrationDto){
  return await this.userService.register(data);
}

@Post('user/login')
async doUserLogin(@Body() data: LoginUserDto, @Req() req: Request): Promise<ApiResponse | LoginInfoDto> {
  const user = await this.userService.getByEmail(data.email)

  if (!user){
    return new Promise(resolve => {
      resolve(new ApiResponse('error', -3001))
    })
  }
}
```

neke od provera tokena:

```
let jwtRefreshData: JwtRefreshDataDto;

try {
    jwtRefreshData = jwt.verify(data.token, jwtSecret);
} catch (e) {
    throw new HttpException('Bad token found', HttpStatus.UNAUTHORIZED);
}

if (!jwtRefreshData) {
    throw new HttpException('Bad token found', HttpStatus.UNAUTHORIZED);
}

if (jwtRefreshData.ip !== req.ip.toString()) {
    throw new HttpException('Bad token found', HttpStatus.UNAUTHORIZED);
}

if (jwtRefreshData.ua !== req.headers["user-agent"]) {
    throw new HttpException('Bad token found', HttpStatus.UNAUTHORIZED);
}
```

*Dopunske informacije*

*Prilozi*

*Pretpostavke i zavisnosti*

*Akronimi i skraćenice*