UNIVERZITET SINGIDUNUM
TEHNIČKI FAKULTET U NOVOM SADU

RAZVOJ INFORMACIONOG SISTEMA ZA PODRŠKU UPRAVLJANJU SPORTSKOM DVORANOM

- diplomski rad -

Mentor: **Kandidat:**

prof. dr ***Đorđe Obradović*** ***Pavle Perić*** *(2017270304)*

Novi Sad, ***2025.***

Sadržaj

- 2 -

Uvod

U savremenom dobu digitalizacije, sve više procesa se prenosi na internet kako bi se poboljšala efikasnost i korisničko iskustvo. Jedno od takvih područja jeste i upravljanje sportskim objektima – dvoranama, salama i terenima koji se svakodnevno koriste za različite sportske aktivnosti. Tradicionalan pristup rezervaciji termina, koji se oslanja na telefonske pozive, lični dolazak ili ručno vođenje evidencije, postaje sve manje održiv, naročito u urbanim sredinama gde je broj korisnika i termina značajan.

Rezervacija sportskih sala često uključuje niz koraka: proveru dostupnosti termina, kontakt sa osobljem, potvrdu rezervacije, kao i eventualno plaćanje. Ovakav pristup može biti spor, neprecizan i sklon greškama. Istovremeno, korisnici očekuju brza, transparentna i automatizovana rešenja koja omogućavaju da u svega nekoliko koraka obave sve što im je potrebno – od pretrage termina do potvrde i plaćanja.

Cilj ovog rada je razvoj web aplikacije koja će korisnicima omogućiti da na jednostavan i pregledan način rezervišu termine za korišćenje sportskih dvorana. Aplikacija treba da omogući registrovanim korisnicima da pretražuju dostupne termine, rezervišu željene termine, upravljaju svojim nalozima i izvrše plaćanje. S druge strane, administratori sistema treba da imaju mogućnost upravljanja salama, pregledom rezervacija i korisnika, kao i osnovne statistike.

Web aplikacija će biti izrađena korišćenjem **Spring Boot** okvira na strani servera (backend), **Angular** okvira za klijentski deo (frontend), dok će se za čuvanje podataka koristiti **PostgreSQL** baza podataka. Online plaćanja omogući nam implementacija **Stripe sistem za plaćanje.** Na ovaj način koristiće se savremene tehnologije koje omogućavaju modularnost, sigurnost i dobru skalabilnost sistema. Pored same implementacije, u radu će biti prikazani i modeli sistema putem **UML dijagrama**, kao i analiza poslovnih pravila i zahteva korisnika.

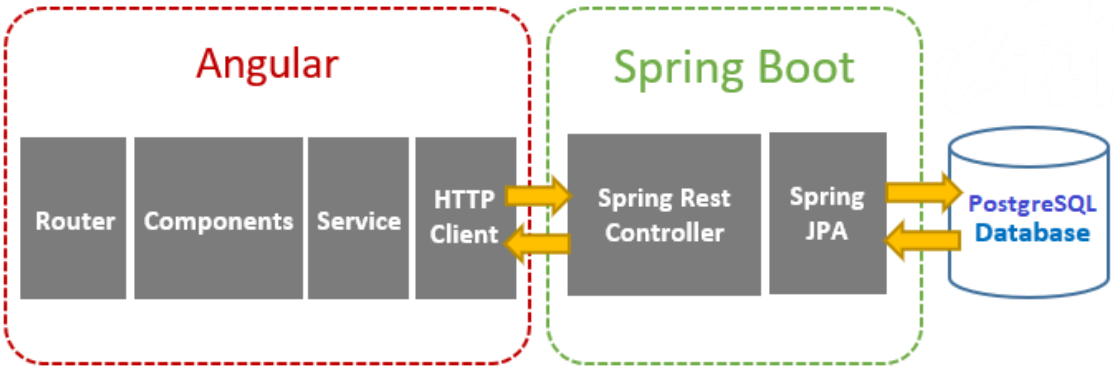
U nastavku rada biće predstavljena specifikacija sistema sa analizom korisnika i funkcionalnosti, zatim tehnička realizacija aplikacije, način testiranja i evaluacije rezultata, kao i zaključci sa predlozima za dalji razvoj i unapređenje aplikacije.

- 3 -

1 Tehnologije i alati korišćeni u razvoju sistema

Razvoj informacionog sistema za upravljanje sportskom dvoranom zahteva primenu pouzdanih i savremenih tehnologija kako bi se obezbedila skalabilnost, sigurnost i efikasnost rada. Odeljak koji sledi pruža kratak pregled tehnologija korišćenih na svim nivoima sistema – backend, frontend, baza podataka i sistemi za integraciju plaćanja.

Na slici 1.1. predstavljena je arhitektura aplikacije sa sledećim tehnologijama. Spring Boot eksportuje REST API-je koristeći Spring Web MVC i komunicira sa PostgreSQL bazom podataka koristeći Spring Data JPA. Angular klijent šalje HTTP zahteve i preuzima HTTP odgovore koristeći HttpClient modul, prikazuje podatke o komponentama. Takođe koristimo Angular ruter za navigaciju do stranica. O arhitekturi projekta detaljnije u potpoglavljima.



Slika 1.1 – Grafički prikaz arihtekture aplikacije

1.1 Backend

Spring Boot predstavlja proširenje okvira Spring Framework, kreiran u cilju pojednostavljenja razvoja enterprise aplikacija u programskom jeziku Java. Razvijen od strane VMware tima, Spring Boot omogućava brzo pokretanje i konfiguraciju aplikacija uz minimalnu količinu ručne konfiguracije. Ključna karakteristika ovog okvira jeste tzv. "auto-configuration" mehanizam, koji automatski detektuje zavisnosti i postavlja inicijalne vrednosti konfiguracije, čime se značajno skraćuje vreme razvoja i smanjuje prostor za greške [1].

Spring Boot omogućava jednostavnu integraciju sa bazama podataka, autentifikaciju i autorizaciju putem Spring **Security** modula, kao i **ORM** pristup kroz Spring **Data JPA**, što programeru omogućava rad sa bazom koristeći objektnu reprezentaciju. Aplikacije izrađene pomoću Spring Boot-a poseduju ugrađeni web server, čime se eliminiše potreba za spoljnim kontejnerima, što je naročito korisno u kontekstu mikrouslužne arhitekture i kontejnerizacije (npr. Docker).

- 4 -

1.2 Frontend

Angular je open-source razvojni okvir za izradu jednostranih veb-aplikacija (eng. *Single Page Applications* – SPA) razvijen od strane kompanije Google. Baziran na programskom jeziku **TypeScript** (nadskup JavaScript-a), Angular omogućava deklarativni pristup razvoju korisničkog interfejsa koristeći komponente, šablone i dvosmerno vezivanje podataka (eng. *two-way data binding*) [2].

Angular koristi tzv. *reactive programming paradigmu* uz pomoć RxJS biblioteke, što omogućava efikasnu obradu asinhronih događaja i upravljanje podacima u realnom vremenu. Okvir podržava modularnu arhitekturu, testabilnost i hijerarhiju komponenti, što ga čini pogodnim za izgradnju skalabilnih web rešenja. U okviru ovog projekta korišćen je i dodatak Angular **Material**, koji implementira dizajnerske smernice.

1.3 Baza podataka

PostgreSQL je moćan, open-source sistem za upravljanje relacionim bazama podataka (RDBMS) koji se aktivno razvija više od tri decenije. Poznat je po svojoj stabilnosti, bezbednosti i podršci za napredne funkcionalnosti kao što su transakcije, indeksiranje, replikacija, kao i rad sa nestrukturiranim podacima putem JSONB formata. Za razliku od jednostavnijih rešenja kao što su MySQL ili SQLite, PostgreSQL pruža napredne mogućnosti prilagođene zahtevnijim sistemima, kao što su složeni upiti, procedure, trigere i poglede[4].

U kontekstu ove aplikacije, PostgreSQL je korišćen za čuvanje korisničkih podataka, rezervacija termina i istorije plaćanja. Integracija sa Spring Boot okvirom realizovana je putem JPA (eng. *Java Persistence API*) sloja, omogućavajući rad sa objektno-relacionim mapiranjem (ORM).

1.4 Stripe

Stripe je napredna platforma za procesuiranje online plaćanja, namenjena programerima i integratorima sistema koji žele da implementiraju sigurne i brze transakcije putem interneta. Stripe omogućava povezivanje sa različitim vrstama platnih kartica, kao i kreiranje sesija plaćanja, tokenizaciju kartičnih podataka i rad sa webhook mehanizmima za praćenje statusa transakcije [3].

Stripe funkcioniše po principu RESTful API integracije, gde backend aplikacija kreira zahtev za uplatu, dok korisnik završava plaćanje na frontend strani, nakon čega se rezultat automatski vraća aplikaciji. Jedna od ključnih karakteristika Stripe-a je **PCI-DSS** kompatibilnost, što znači da aplikacija nikada ne rukuje direktno podacima o karticama, čime se znatno povećava bezbednost sistema.

Stripe podržava i napredne funkcionalnosti kao što su plaćanje u više valuta, automatsko fakturisanje, integracija sa mobilnim aplikacijama, kao i mehanizmi za povrat novca i otkazivanje transakcija.

- 5 -

1.5 Alati

Pod ostalim alatima koje uključuju ceo proces izrade aplikaje jesu:

- **Visual Studio Code** – Moćan i široko korišćen integrisani razvojni alat (IDE) koji se koristi za razvoj frontend aplikacija. Zbog svoje fleksibilnosti, podrške za mnoge programske jezike i velikog broja dostupnih dodataka, predstavlja jedan od najpopularnijih editora za rad sa Angular, React i drugim frontend tehnologijama.
- **Git** – Sistem za verzionu kontrolu koda koji omogućava praćenje promena, upravljanje verzijama i timski rad na softverskim projektima. U ovom radu koristi se za upravljanje backend kodom, olakšavajući kolaboraciju i praćenje izmjena tokom razvoja.
- **Postman** – Postman je alat namenjen testiranju i razvoju API-ja. Omogućava kreiranje, slanje i proveru HTTP zahteva, što je od ključne važnosti za proveru funkcionalnosti backend servisa i njihovu integraciju sa frontend delom aplikacije.
- **Draw.io** – Alat za kreiranje dijagrama i vizuelnih prikaza sistema. Korišćen je za planiranje arhitekture aplikacije, modelovanje procesa i grafičko predstavljanje ključnih komponenti softverskog rešenja. To ujedno jeste i prvi korak u planiranju i kreiranju strukture projekta, pa i ujedno i njegove baze podataka. O nacrtanim dijagramima sa svim objašnjenjima šta oni predstavljalju više u sledećem podglavlju.

- 6 -

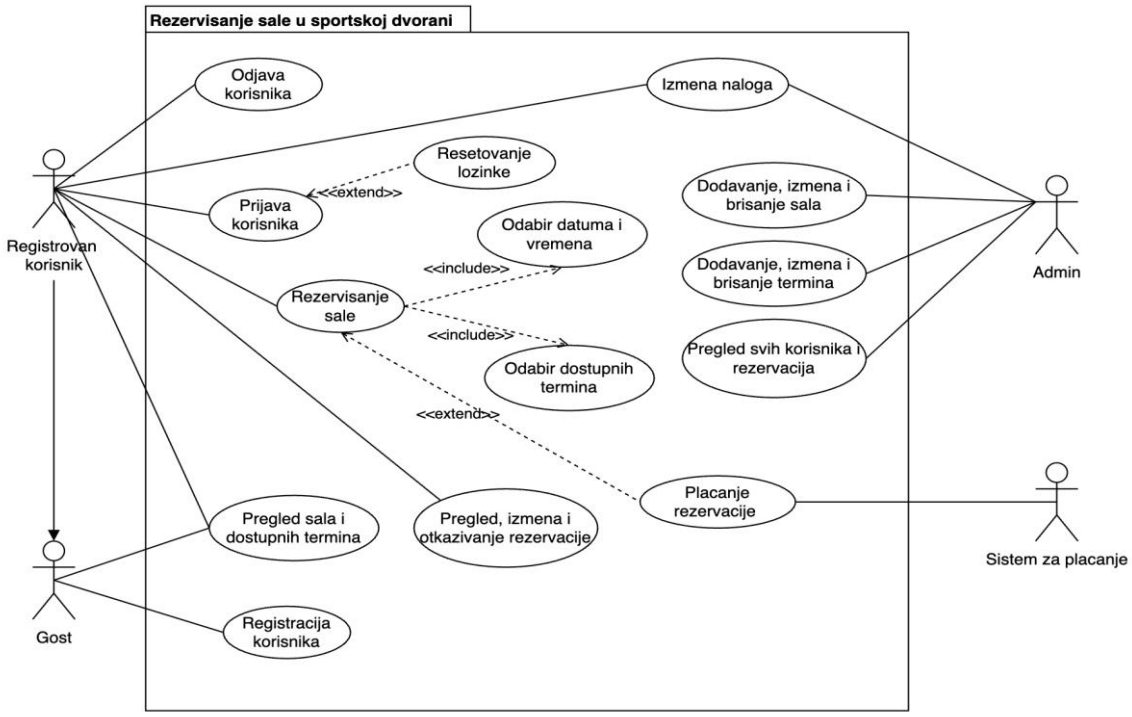
2 Analiza korisničkih zahteva

U ovom poglavlju biće predstavljeni korisnički zahtevi koji bi trebalo da budu implementirani u sistemu i oni će biti predstavljeni pomoću dijagrama slučajeva upotrebe i dijagram klasa, koji prikazuje klase koje obuhvataju ceo sistem.

2.1 Dijagram slučajeva upotrebe

Tokom analize zahteva, dijagrami slučaja upotrebe pomažu da se identifikuju akteri i da se definiše ponašanje sistema [14]. Modeliranje slučaja upotrebe je prihvaćeno i široko se koristi u industriji. Pored aktera i slučajeva upotrebe UML definiše mali skup odnosa za strukturu aktera i slučajeve upotrebe. Slučajevi korišćenja mogu biti povezani sa drugim slučajevima korišćenja sledećim odnosima:

1. <<**extend**>> – odnos **proširenja** podrazumeva da slučaj upotrebe može da proširi ponašanje opisano u drugom slučaju upotrebe, kojim upravlja uslov
2. <<**include**>> – odnos **uključivanja** znači da slučaj upotrebe uključuje ponašanje opisano u drugom slučaju upotrebe
3. **Generalization** – odnos generalizacije znači da je dete specifičniji oblik roditeljskog slučaja upotrebe. Dete nasleđuje sve karakteristike i asocijacije roditelja i može da dodaje nove karakteristike i udruženja [15]



2.1.1 Slučaj upotrebe: Logovanje

Kratak opis: Prijava korisnika na sistem.

- 7 -

Učesnici: Registrovani korisnik.

Preduslovi: Korisnik mora imati kreiran nalog i uneti tačno korisničko ime i lozinku.

Osnovni tok događaja:

1. Korisnik otvara stranicu za prijavu.
2. Korisnik unosi korisničko ime i lozinku.
3. Sistem proverava kredencijale.
4. Ukoliko su podaci tačni, korisnik se uspešno prijavljuje na sistem.
5. Ako korisnik nema nalog, može se preusmeriti na stranicu za registraciju.
6. Ako je zaboravljena lozinka, korisnik može zatražiti resetovanje putem mejla.

Alternativni tokovi:

- **Resetovanje lozinke:**
  - Korisnik klikne na link za resetovanje lozinke.
  - Unosi svoju mejl adresu.
  - Ako mejl postoji u sistemu, dobija uputstva za reset lozinke.

Postuslovi: Korisnik je uspešno prijavljen na sistem. Uspešno se odjavljuje sa svog naloga i prekida sesiju u sistemu.

2.1.2 Slučaj upotrebe: Odjavljivanje

Kratak opis: Korisnik se odjavljuje sa svog naloga i prekida sesiju u sistemu.

Učesnici: Registrovani korisnik.

Preduslov: Korisnik je uspešno prijavljen na sistem.

Osnovni tok:

1. Klik na dugme za odjavu.
2. Sistem prekida sesiju.
3. Korisnik se vraća na početnu stranicu.

Alternativni tokovi: Nema.

Postuslov: Korisnik je odjavljen.

- 8 -

2.1.3 Slučaj upotrebe: Registracija korisnika

Kratak opis: Gost kreira korisnički nalog.

Učesnici: Gost.

Preduslovi: Gost nije prethodno registrovan.

Osnovni tok:

1. Gost otvara stranicu za registraciju.
2. Popunjava formular sa traženim podacima.
3. Sistem validira podatke.
4. Po uspešnoj validaciji, kreira se novi korisnički nalog.

Alternativni tokovi: Nema.

Postuslov: Korisnik je uspešno registrovan.

2.1.4 Slučaj upotrebe: Pregled sala i dostupnih termina

Kratak opis: Gost ili registrovani korisnik vidi sve sale i dostupne termine.

Učesnici: Gost i registrovani korisnik.

Preduslovi: Nema.

Osnovni tok:

1. Korisnik se usmerava na stranicu za pregled svih sala i dostupnih termina
2. Sistem prikazuje listu sala sa slobodnim terminima.

Alternativni tokovi: Nema.

Postuslovi: Korisnik ima uvid u mogućnosti rezervacije.

2.1.5 Slučaj upotrebe: Rezervisanje sale

Kratak opis: Registrovani korisnik rezerviše salu u odabranom terminu.

Učesnici: Registrovani korisnik i Stripe sistem za plaćanje.

Preduslovi: Korisnik mora biti prijavljen.

Osnovni tok:

- 9 -

1. Korisnik bira salu.
2. Bira datum i vreme za rezervaciju.
3. Pregleda dostupne termine.
4. Potvrđuje i kreira rezervaciju.

Alternativni tokovi:

- Korisnik može samo da rezerviše i plati uživo svoj termina, a to može i da uradi odmah, online putem.
- Ako je potvrdio rezervaciju može odmah i da plati putem stripe sistema.

Postuslovi: Rezervacija je uspešno zabeležena.

2.1.6 Slučaj upotrebe: Pregled, izmena i otkazivanje rezervacije

Kratak opis: Korisnik menja ili otkazuje postojeću rezervaciju.

Učesnici: Registrovani korisnik.

Preduslovi: Korisnik mora imati prethodno kreiranu rezervaciju.

Osnovni tok:

1. Korisnik pregleda svoje rezervacije.
2. Po izboru, može da ih izmeni ili otkaže.
3. Ako želi da izmeni, može samo da izbare u tom momentu dostupne termine.

Postuslovi: Rezervacija je izmenjena ili otkazana.

2.1.7 Slučaj upotrebe: Plaćanje rezervacije

Kratak opis: Korisnik vrši plaćanje rezervacije.

Učesnici: Registrovani korisnik i Stripe sistem za plaćanje

Preduslovi: Postoji validno kreirana rezervacija i validan korisnik u sistemu.

Osnovni tok:

1. Korisnik pristupa opciji za plaćanje.
2. Unosi potrebne podatke.
3. Sistem za plaćanje obrađuje transakciju.
4. Dobija potvrdu o uspešnoj uplati.

Alternativni tokovi: Nema.

- 10 -

Postuslovi: Rezervacija je uspešno nasplaćena.

2.1.8 Slučaj upotrebe: Izmena naloga

Kratak opis: Korisnik menja svoje podatke (npr. lozinku, ime, e-mail).

Učesnici: Registrovani korisnik i admin

Preduslovi: Korisnik ima nalog i prijavljen je u sistem.

Osnovni tok:

1. Korisnik ili admin pristupa stranici za izmenu naloga.
2. Menja željene podatke.
3. Kada završi klikom na dugme sačuvaće svoje izmenjene podatke.
4. Sistem potvrđuje i pamti izmenu.

Alternativni tokovi: Nema.

Postuslovi: Podaci o nalogu korisnika ili admina su ažurirani.

2.1.9 Slučaj upotrebe: Upravljanje sportskim salama i terminima za rezervaciju

Kratak opis: Administrator ima mogućnost upravljanja salama kao i terminima za korišćenje sportskih sala, što podrazumeva dodavanje novih termina ili sala, izmenu postojećih kao i njihovo brisanje ukoliko više nisu dostupni ili važeći.

Učesnici: Admin.

Preduslovi: Admin mora biti prijavljen na sistem sa odgovarajućim ovlašćenjima.

Osnovni tok događaja:

1. Administrator pristupa interfejsu za upravljanje dvoranom.
2. Ulazi na karticu sa listom sala ili termina.
3. Ima mogućnost da:
    a. Dodaje nove termine dostupne za rezervaciju ili nove sale,
    b. Ažurira vreme postojećih termina ili bilo koje druge podatke vezane za termine ili sale,
    c. Obriše termine koji više nisu dostupni ili su zastareli ili sale.

Alternativni tokovi: Nema.

Postuslovi: Termini za odabranu salu su uspešno ažurirani i dostupni korisnicima u realnom vremenu. Isti princip i za salu, izvršen uspešno i baza je ažurirana.

- 11 -

2.1.10 Slučaj upotrebe: Pregled korisnika i njihovih rezervacija

Kratak opis: Administrator ima uvid u sve registrovane korisnike sistema, kao i pregled svih njihovih izvršenih i zakazanih rezervacija sala.

Učesnici: Admin.

Preduslovi: Admin je uspešno prijavljen na sistem i ima odgovarajuće administratorske privilegije.

Osnovni tok događaja:

1. Administrator pristupa administrativnoj kontrolnoj tabli.
2. Pokreće prikaz liste korisnika sistema.
3. Za svakog korisnika može da pregleda:
    a. Lične podatke (npr. ime, e-mail),
    b. Istoriju i status rezervacija (potvrđene, otkazane, zakazane)

Alternativni tokovi: Nema.

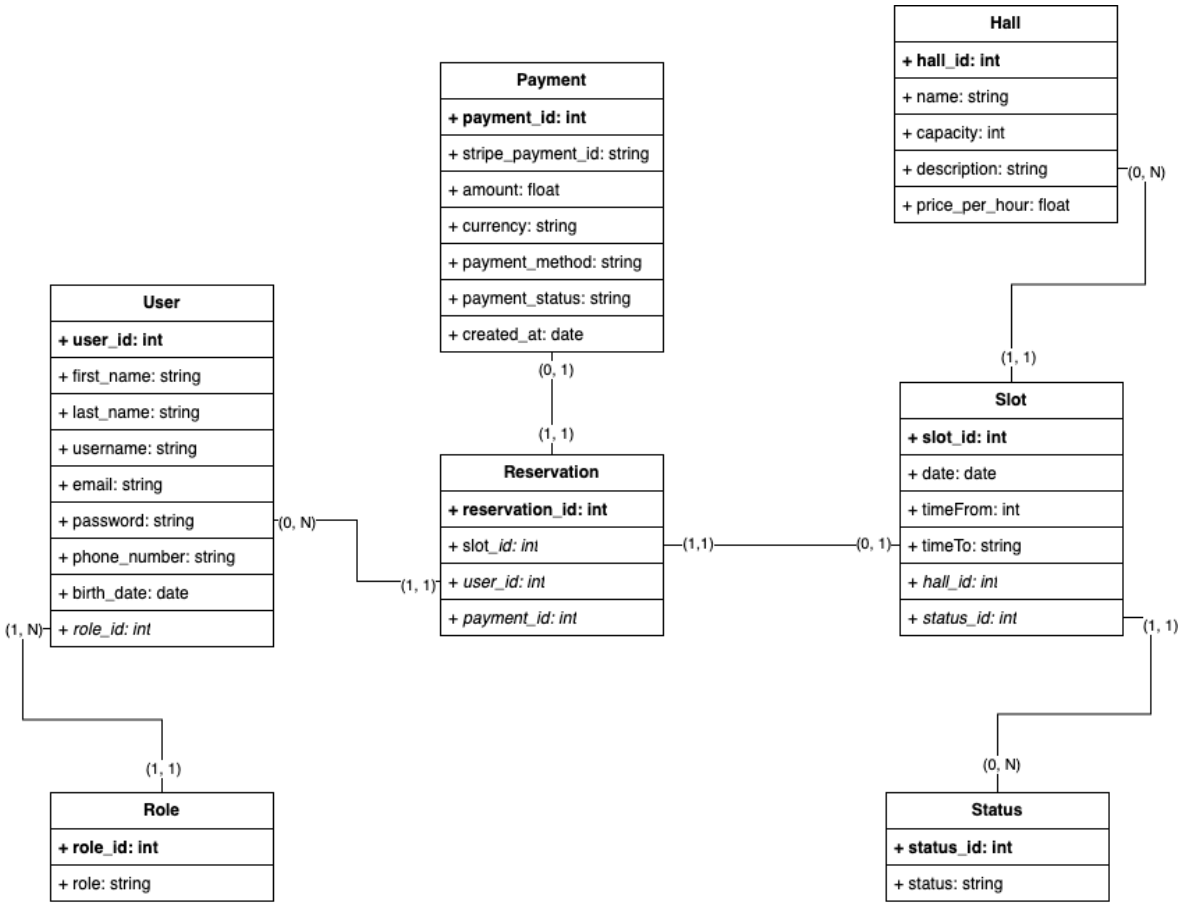
Postuslovi: Administrator je dobio kompletan i ažuriran uvid u korisničke naloge i pripadajuće rezervacije, što omogućava efikasno upravljanje i nadzor nad radom sistema.

2.2 Dijagram klasa

Dijagram klasa je glavni gradivni blok objektno-orijentisanog modelovanja. Koristi se za opšte konceptualno modelovanje strukture aplikacije, kao i za detaljno modelovanje prevođenja modela u programski kod. Dijagrami klasa mogu se koristiti i za modelovanje podataka. Klase u dijagramu klasa predstavljaju glavne elemente, interakcije u aplikaciji i klase koje treba da budu isprogramiranje.

U slučaju informacionog sistema za upravljanje sportskom dvoranom, dijagram klasa se sastoji iz sledećih klasa: User, Role, Hall, Slot, Reservation, Status, Payment. U nastavku sledi analiza svake od klasa pojedinačno.

- 12 -



Dijagram 2: Dijagram klasa

1. **User** - Ova klasa predstavlja korisnike sistema, koji mogu imati različite uloge u zavisnosti od povezanosti sa klasom Role. Svaki korisnik može kreirati više rezervacija, a može biti i administrator u zavisnosti od dodeljene uloge.

   - **user_id:** Jedinstveni identifikator korisnika (primarni ključ).
   - **first_name:** Ime korisnika.
   - **last_name:** Prezime korisnika.
   - **username:** Korisničko ime, obelezje korisnikovog naloga.
   - **email:** Imejl adresa korisnika, koristi se za prijavu.
   - **password:** Lozinka korisnika, koja se čuva u enkriptovanom obliku.
   - **phone_number:** Broj telefona korisnika.
   - **birth_date:** Datum rođenja korisnika.
   - **role_id:** Strani ključ koji povezuje korisnika sa njegovom ulogom u sistemu.

2. **Role** - Klasa Role definiše uloge koje korisnici mogu imati u sistemu. Ovo omogućava diferencijaciju prava pristupa i funkcionalnosti između, administratora i registrovanog člana.
   - **role_id:** Jedinstveni identifikator uloge (primarni ključ).
   - **role**: Naziv uloge, npr. “admin”, , “user”.

- 13 -

3. **Hall –** Klasa Hall predstvalja jednu salu u okviru sportske dvorane koja se može rezervisati.
   - **hall_id:** Jedinstveni identifikator sale (primarni ključ).
   - **title**: Naziv sale.
   - **capacity**: Maksimalan broj korisnika koji sala može da primi.
   - **description:** Tekstualni opis sale (npr. veličina, oprema, tip sportova).
   - **price_per_hour:** Cena korišćenja sale po satu.

4. **Slot –** Klasa Slot predstavlja konkretan termin koji korisnik može rezervisati.
   - **slot_id:** Jedinstveni identifikator termina (primarni ključ).
   - **date**: Datum održavanja termina.
   - **start_time**: Vreme početka termina.
   - **end_time**: Vreme završetka termina.
   - **hall_id:** Strani ključ koji pokazuje kojoj sali termin pripada.
   - **status_id:** Strani ključ koji pokazuje status termina.

5. **Status –** Klasa Status određuje stanje termina, što omogućava filtriranje i upravljanje raspoloživošću.
   - **status_id:** Jedinstveni identifikator statusa (primarni ključ).
   - **status**: Naziv statusa (“available”, “reserved”, “canceled”).

6. **Reservation –** Klasa Reservation beleži informacije o rezervacijama termina. Spaja više tabela i daje konačnu sliku jednog rezervisanog termina od strane jednog korisnika.
   - **reservation_id:** Jedinstveni identifikator rezervacije (primarni ključ).
   - **is_payed**: Informacija da li je rezervacija plaćena (true ili false). Pošto korisnik će moći da izvrši plaćanje online ili uživo na šalteru sportske dvorane.
   - **user_id:** Strani ključ koji pokazuje koji korisnik je izvršio rezervaciju.
   - **slot_id:** Strani ključ koji pokazuje na koji termin se odnosi rezervacija.
   - **hall_id:** Strani ključ koji povezuje rezervaciju sa konkretnom salom.

7. **Payment** - Klasa Payment evidentira podatke o transakcijama korisnika. Omogućava uvid u naplatu termina.
   - **payment_id:** Jedinstveni identifikator plaćanja (primarni ključ).
   - **amount**: Iznos plaćen za rezervaciju.
   - **payment_date**: Datum kada je izvršeno plaćanje.
   - **payment_status**: Status plaćanja (npr. “Uspelo”, “Neuspešno”).
   - **reservation_id:** Strani ključ – za koju rezervaciju je izvršeno plaćanje.
   - **user_id:** Strani ključ – ko je izvršio plaćanje.

Međusobne veze između tabela:

- **User / Role –** Svaki korisnik ima tačno 1 ulogu, dok 1 uloga može da bude dodeljena više korisnika.
- **Reservation / User –** Svaka rezervacija pripada tačno 1 korisniku, dok 1 korisnik može da napravi više rezervacija

- 14 -

- **Reservation / Payment –** Svako plaćanje pripada tačno 1 rezervaciji, dok 1 rezervacija može, a i ne mora imati plaćanje zbog opcije plaćanja uživo
- **Reservation / Slot –** Svaka rezervacija je vezana za 1 termin, dok 1 termin može, a i ne mora biti vezan za rezervaciju (npr. Kada je u statusu “available”)
- **Slot / Hall –** Svaki termin pripada tačno 1 sali, dok 1 sala može imati više termina
- **Slot / Status –** Svaki termin može da ima 1 status, dok 1 status može biti dodeljen više termina

- 15 -

3 Zaključna razmatranja

U svom istraživanju u ovom radu došao/la sam do sledećih važnih nalaza. Tekst.

Tekst Tekst.

Tekst Tekst.

Tekst Tekst.

(Zaključak treba da bude efektan. U njemu ne treba prepričavati ono što je ranije rečeno, već treba sumirati najvažnije i dati preporuke za dalja istraživanja tog problema. Po dužini, zaključak treba da odgovara dužini uvoda).

- 16 -

Literatura

1. Spring IO. (n.d.) *Spring Boot – Official Documentation* [online]. Dostupno na: https://spring.io/projects/spring-boot
2. Angular. (n.d.) *Angular – Official Website* [online]. Dostupno na: https://angular.io
3. Stripe. (n.d.) *Online Payment Platform – Stripe Documentation* [online]. Dostupno na: https://stripe.com/docs
4. PostgreSQL Global Development Group. (n.d.) *PostgreSQL Documentation* [online]. Dostupno na: https://www.postgresql.org/docs/

- 17 -