

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 124

**BAZA PODATAKA I WEB-APLIKACIJA ZA
VELEPRODAJNU TVRTKU**

Luka Ilić

Zagreb, lipanj 2021.

ZAVRŠNI ZADATAK br. 124

Pristupnik: **Luka Ilić (0036514047)**
Studij: Elektrotehnika i informacijska tehnologija i Računarstvo
Modul: Računarstvo
Mentor: prof. dr. sc. Boris Vrdoljak

Zadatak: **Baza podataka i web-aplikacija za veleprodajnu tvrtku**

Opis zadatka:

U sklopu ovog završnog rada potrebno je izraditi bazu podataka u koju će se pohranjivati podaci o zaposlenicima veleprodajne tvrtke, kupcima, proizvodima, stanju na skladištu, narudžbama i računima. Nakon oblikovanja modela entiteti-veze i odgovarajućeg relacijskog modela baze podataka, treba implementirati bazu podataka koristeći sustav za upravljanje bazom podataka PostgreSQL. Potrebno je zatim korištenjem programskog jezika Java napraviti web-aplikaciju koja korisnicima omogućava unos, izmjenu, pregled i pretraživanje podataka.

Rok za predaju rada: 11. lipnja 2021.

Sadržaj

| | |
|------------------------------------|----|
| Uvod..... | 1 |
| 1. Specifikacija zahtjeva..... | 2 |
| 2. Baza podataka | 4 |
| 2.1. Korištene tehnologije | 4 |
| 2.2. Opis domene..... | 4 |
| 2.3. ER model..... | 5 |
| 2.3.1. Entiteti..... | 6 |
| 2.3.2. Veze..... | 7 |
| 2.4. Relacijski model..... | 8 |
| 2.5. SQL naredbe..... | 9 |
| 2.5.1. Stvaranje tablica | 9 |
| 2.5.2. Okidači | 10 |
| 3. Web aplikacija..... | 13 |
| 3.1. Korištene tehnologije | 13 |
| 3.2. Arhitektura | 14 |
| 3.3. Implementacija..... | 14 |
| 3.3.1. Backend..... | 14 |
| 3.3.2. Frontend | 19 |
| 3.4. Upute za korištenje..... | 23 |
| 3.4.1. Zaposlenik u skladištu..... | 23 |
| 3.4.2. Direktor tvrtke..... | 30 |
| 3.4.3. Administrator | 35 |
| Zaključak..... | 37 |
| Literatura | 38 |
| Sažetak | 39 |

Summary 40

Uvod

Svaka tvrtka koja se bavi nekim oblikom trgovine treba sustav za praćenje stanja skladišta, zaprimanja i izdavanja robe, podataka o proizvodima, poslovnim partnerima te svojim zaposlenicima. Također, osim pohrane i pregleda podataka, iznimno je bitan uvid u poslovanje i statistička analiza prodaje kako bi se uočili nedostaci te maksimalizirao profit. Osim mnogih komercijalnih rješenja za velike korporacije, brojne manje tvrtke se odlučuju za naručivanje takvih sustava s vlastitim specifičnim zahtjevima. Potaknut time i radnim iskustvom u takvoj tvrtki, odlučio sam oblikovati sustav s navedenim funkcionalnostima.

Implementacija sustava sastoji se od baze podataka i web aplikacije, a omogućene su različite funkcionalnosti ovisno o ulozi, odnosno radnom mjestu zaposlenika. Korisničko sučelje izrađeno je s ciljem jednostavnosti, intuitivnosti, ubrzanja poslovanja i minimalnim redundancijama.

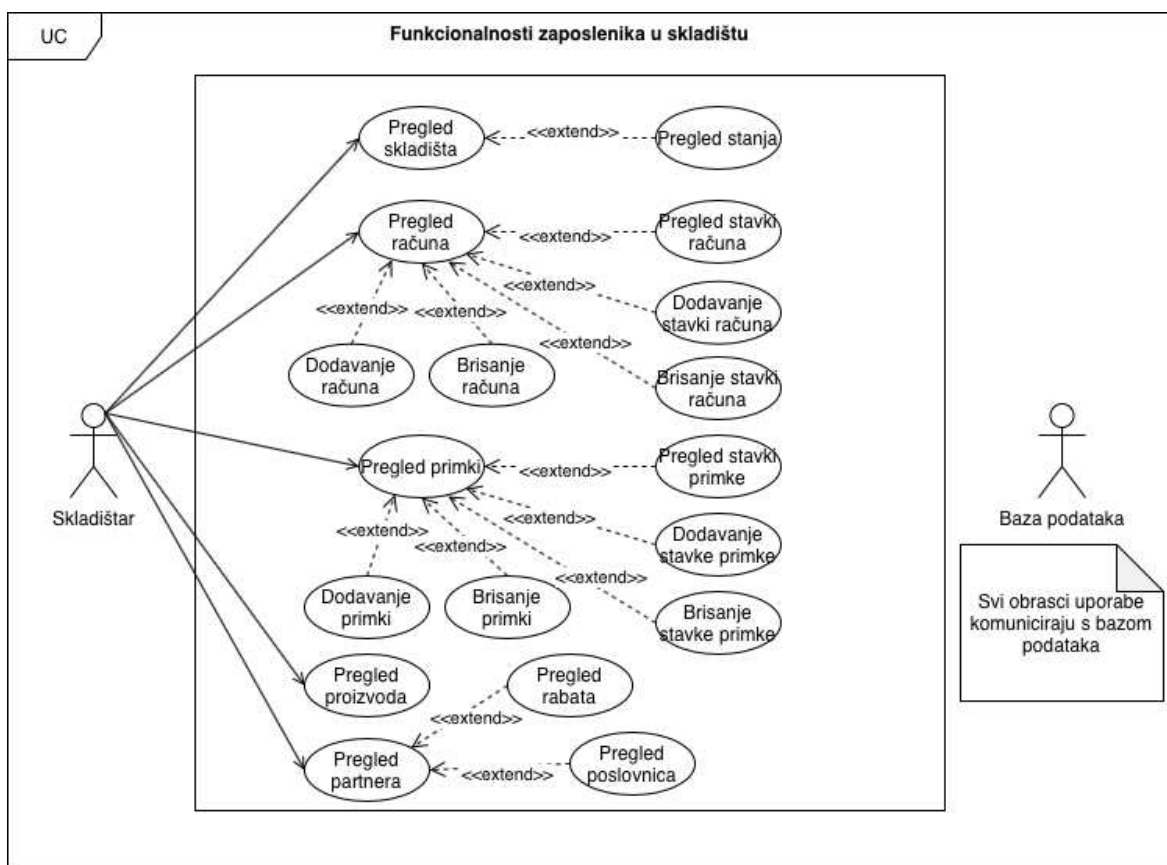
Prvi dio rada opisuje funkcionalne zahtjeve. Drugi dio sadrži postupak modeliranja i implementacije baze podataka, a u trećem se iznose detalji web aplikacije, njezine arhitekture, postupak izgradnje te pregled različitih funkcionalnosti po ulogama zaposlenika.

1. Specifikacija zahtjeva

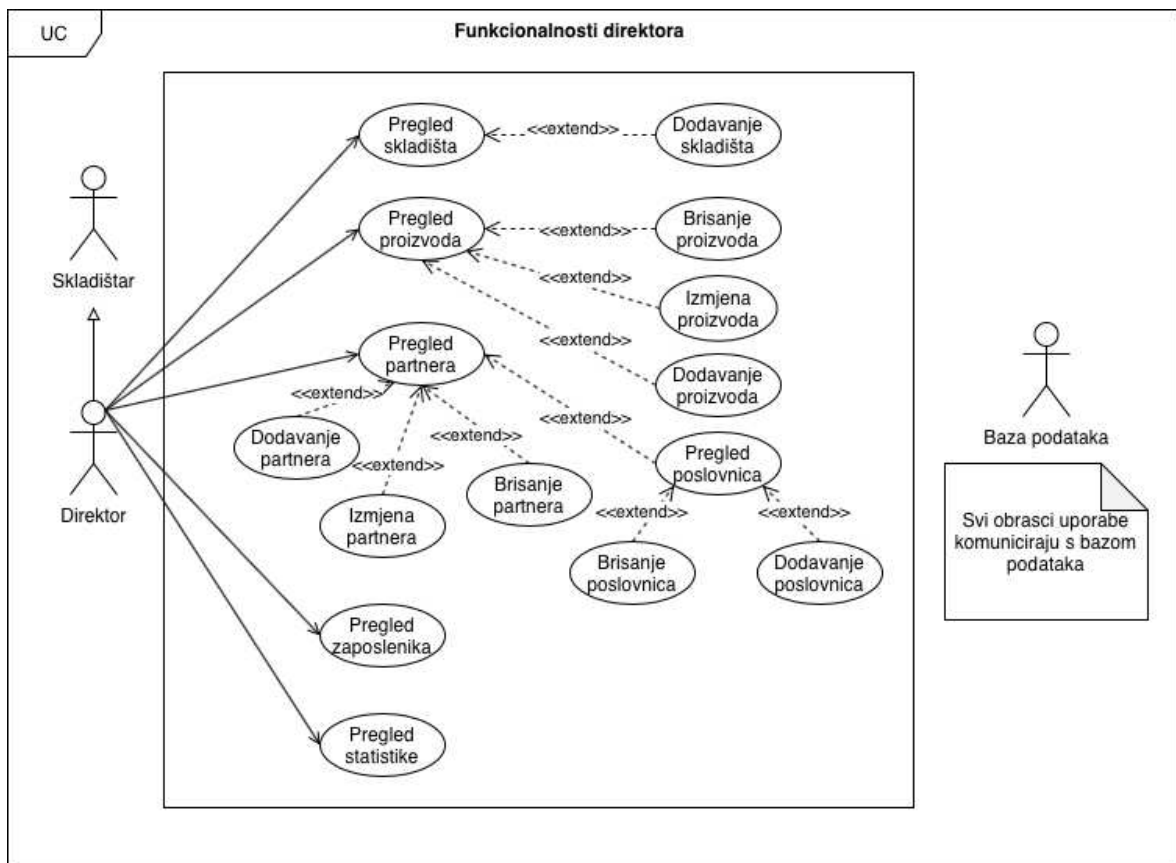
Prije modeliranja ostatka sustava utvrđeni su glavni dionici:

- Zaposlenik u skladištu
- Direktor
- Administrator

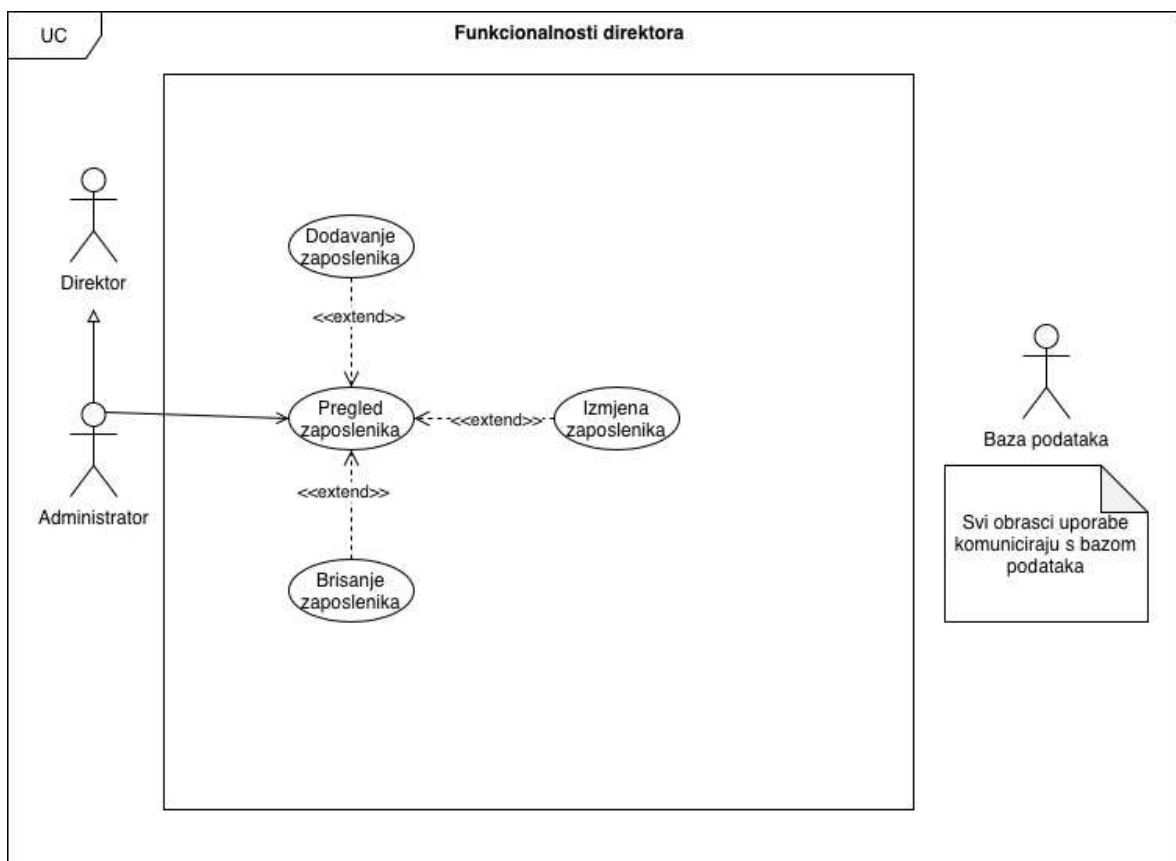
Potom su funkcionalni zahtjevi poput pregledavanja, dodavanja, izmjenjivanja i brisanja podataka te pregleda statistike poslovanja opisani i raspoređeni prema aktorima. Prikazani su na UML dijagramima (engl. *Unified Modeling Language*) obrazaca uporabe (Sl. 1.1, Sl. 1.2, Sl. 1.3), a detaljno su opisani u trećem poglavlju kroz upute za korištenje web aplikacije.



Sl. 1.1 UML dijagram obrazaca uporabe zaposlenika u skladištu



Sl. 1.2 UML dijagram obrazaca uporabe direktora



Sl. 1.3 UML dijagram obrazaca uporabe administratora

2. Baza podataka

2.1. Korištene tehnologije

PostgreSQL

Odabran jer je popularan besplatan sustav za upravljanje bazama podataka, otvorenog koda, dostupan na svim većim operacijskim sustavima te s odličnom reputacijom. Korišten u kombinaciji s alatom pgAdmin za lakši pristup putem korisničkog sučelja kroz internetski pretraživač.

Draw.io

Popularan i koristan alat za crtanje brojnih vrsta dijagrama, pristupa mu se putem internetskog pretraživača te ima mogućnost izvoza dijagrama u odabranom slikovnom formatu u zadanoj rezoluciji.

ERDPlus

Koristan alat za generiranje relacijskog modela iz nacrtanog ER modela, definiranje ograničenja i tipova podataka. Također, postoji opcija generiranja SQL naredbi za stvaranje tablica iz dobivenog relacijskog modela. Pristupa mu se putem internetskog pretraživača, a veći nedostatak mu je što ne podržava trostruke veze.

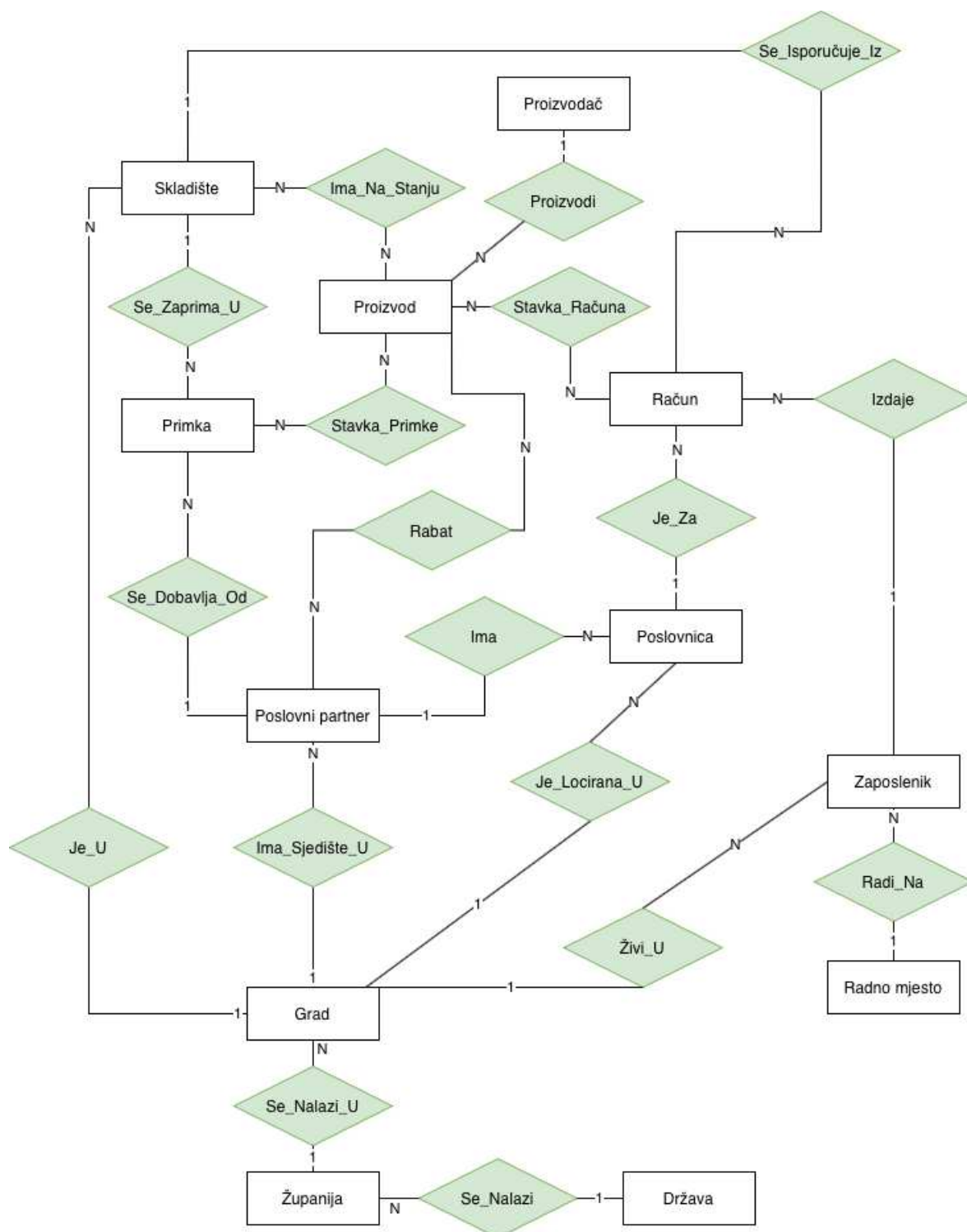
2.2. Opis domene

Pošto je baza modelirana za praćenje rada tvrtke koja se bavi veleprodajom, logički se proizvod nalaže kao važan entitet za koji će se evidentirati ime, prodajna cijena, valuta te cijene, masa, proizvođač, bar kod, količina u kutiji, vrsta te podgrupa koja će detaljnije označavati liniju ili grupu proizvoda određenog proizvođača. Proizvodi se nalaze u skladištu za koje se evidentira adresa, grad i veličina, odnosno površina u metrima kvadratnim. Između proizvoda i skladišta je N-N veza kojom će se bilježiti stanje proizvoda na skladištu. Za sve zaposlenike tvrtke prati se njihovo ime, prezime, email adresa, telefonski broj, grad stanovanja, datum rođenja, datum početka i završetka rada te radno mjesto na kojem su zaposleni. Poduzeće surađuje s poslovnim partnerima, a evidentiraju se njihovi nazivi, adrese, telefonski brojevi, ugovorena odgoda plaćanja u danima, rabat za svaki proizvod te u kojem gradu im je sjedište. Za svaki grad bilježi se

naziv te u kojoj županiji, odnosno državi se nalazi. Poslovni partneri imaju poslovnice. Iz poslovnica dolaze narudžbe koje se unose kao računi. Za svaki račun prati se broj računa, datum, valuta, iz koje je poslovnice došla narudžba, iz kojeg skladišta se izdaje roba, koji zaposlenik ga je izdao te je li račun plaćen. Također, postoji N-N veza s proizvodom kojom će se evidentirati stavke računa, odnosno količinama proizvoda tog računa. Ukoliko su pohranjene prodajne cijene proizvoda u različitim valutama preračunat će se u valutu računa. Izdavanjem računa smanjuje se stanje proizvoda na skladištu. Suprotno tome, zaprimanja robe od dobavljača evidentiraju se primkama. Za primke se bilježi datum, valuta, u koje skladište dolazi roba, od kojeg poslovnog partnera je dobavljena te analogno računu, N-N vezom s proizvodom bilježe se količine zaprimljenih proizvoda i po kojoj cijeni su nabavljeni.

2.3. ER model

Najčešće korištena konceptualna shema za opis baze podataka. Sadrži osnovne i najbitnije podatke o bazi podataka, a to su entiteti, veze među njima te kardinalnost tih veza. Iz opisa domene napravljen je ER model vidljiv na dijagramu (Sl. 2.1).



Sl. 2.1 ER dijagram baze podataka

2.3.1. Entiteti

PROIZVOD – sifProizvod, nazivProizvod, barkod, cijena, valuta, mjera, vrsta, podgrupa, kolicinaUkutiji, masa

SKLADISTE – sifSkladiste, nazivSkladiste, površina

ZAPOSLENIK – sifZaposlenik, oibZaposlenik, ime, prezime, email, password, brojTel, datumRod, datumPoc, datumKraj

POSLOVNIPARTNER – sifPartner, nazivPartner, brojTelPartner, odgoda

POSLOVNICA – sifPoslovnica, nazivPoslovnica, brojTelPoslovnica

RACUN – sifRacun, datumRacun, valutaRacun, placen

PRIMKA – sifPrimka, datumPrimka, valutaPrimka

PROIZVODAC – sifProizvodac, nazivProizvodac

RADNOMJESTO – sifRadnoMjesto, nazivRadnoMjesto

GRAD – sifGrad, nazivGrad

ZUPANIJA – sifZupanija, nazivZupanija

DRZAVA – sifDrzava, nazivDrzava

2.3.2. Veze

Ima_Na_Stanju – sifSkladiste, sifProizvod, kolicinaStanje

Stavka_Racuna – sifRacun, sifProizvod, kolicinaProizvodRacun

Stavka_Primke – sifPrimka, sifProizvod, kolicinaProizvodPrimka, kupovnaCijena

Rabat – sifPartner, sifProizvod, iznosRabat

Se_Zaprima_U – sifPrimka, sifSkladiste

Se_Isporucuje_Iz – sifRacun, sifSkladiste

Se_Dobavlja_Od – sifPrimka, sifPartner

Proizvodi – sifProizvod, sifProizvodac

Je_U – sifSkladiste, sifGrad, adresaSkladiste

Ima_Sjediste_U – sifPartner, sifGrad, adresaPartner

Je_Locirana_U – sifPoslovnica, sifGrad, adresaPoslovnica

Se_Nalazi_U – sifGrad, sifZupanija

Se_Nalazi – sifZupanija, sifDrzava

Zivi_U – sifZaposlenik, sifGrad, adresaStanovanja

Radi_Na – sifZaposlenik, sifRadnoMjesto

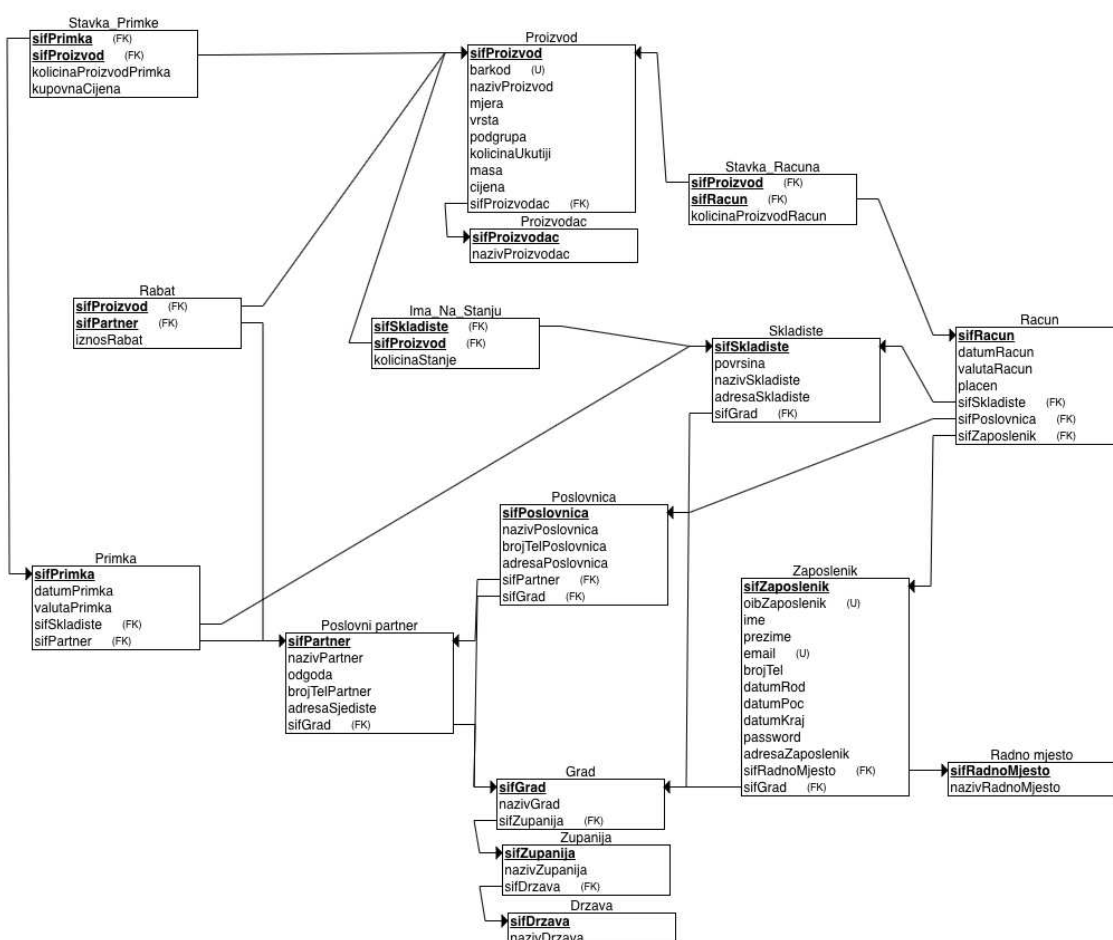
Ima – sifPoslovnica, sifPartner

Je_Za – sifRacun, sifPoslovnica

Izdaje – sifRacun, sifZaposlenik

2.4. Relacijski model

Za dobivanje relacijskog modela iz postojećeg ER modela postoji definirani skup pravila te postoje mnogi alati koji to brzo i točno obavljaju, olakšavajući proces izgradnje baze podataka. Tako je iz prethodno opisanog ER modela korištenjem alata ERDPlus dobiven relacijski model prikazan na slici (Sl. 2.2). Primarni ključevi su podebljani i podcrtani, dok su strani ključevi i UNIQUE ograničenja označena u zagradama.



Sl. 2.2 Relacijski model baze podataka

2.5. SQL naredbe

2.5.1. Stvaranje tablica

Nakon izrade relacijskog modela, u alatu ERDPlus postavljene su sve vrste podataka, te UNIQUE ograničenja na OIB i email zaposlenika te bar kod proizvoda. Potom su generirane SQL naredbe za stvaranje tablica u kojima su svi primarni ključevi promijenjeni iz INT u SERIAL kako bi baza automatski inkrementirala primarne ključeve. Većini atributa ostavljeno je ograničenje NOT NULL jer proizvod bez cijene ili račun bez valute nemaju smisla. Također, dodana su ograničenja za kaskadno brisanje na strane ključeve gdje je to imalo smisla, primjerice na strane ključeve sifRacun u Stavka_Racuna i sifPrimka u Stavka_Primke jer kod brisanja računa slijedi brisanje svih stavki tog računa, dok kod zatvaranja pojedine poslovnice nema smisla obrisati sve račune izdane za tu poslovnicu. Primjer gotove naredbe za kreiranje tablice dan je u nastavku (Kôd 2.1).

```
CREATE TABLE Primka (
    sifPrimka SERIAL NOT NULL,
    datumPrimka DATE NOT NULL,
    valutaPrimka VARCHAR(50) NOT NULL,
    sifSkladiste INT NOT NULL,
    sifPartner INT NOT NULL,
    PRIMARY KEY (sifPrimka),
    FOREIGN KEY (sifSkladiste) REFERENCES
    Skladiste(sifSkladiste),
    FOREIGN KEY (sifPartner) REFERENCES
    Poslovni_partner(sifPartner)
);
CREATE TABLE Stavka_Primke(
    kolicinaProizvodPrimka INT NOT NULL,
    kupovnaCijena FLOAT NOT NULL,
    sifPrimka INT NOT NULL,
    sifProizvod INT NOT NULL,
    PRIMARY KEY (sifPrimka, sifProizvod),
    FOREIGN KEY (sifPrimka) REFERENCES Primka(sifPrimka) ON
    DELETE CASCADE,
    FOREIGN KEY (sifProizvod) REFERENCES Proizvod(sifProizvod)
);
```

Kôd 2.1 Primjer CREATE TABLE naredbi

Nakon stvaranja baze podataka i svih tablica, u svaku od njih su INSERT naredbama dodane n-torke kako bi se u daljnjem razvoju mogla provjeriti funkcionalnost dohvaćanja podataka iz baze, a svi ostali podaci su naknadno uneseni putem funkcionalne web aplikacije.

2.5.2. Okidači

Kako bi se automatiziralo praćenje stanja proizvoda na skladištu te osiguralo domensko ograničenje da stanje niti jednog proizvoda ne smije biti manje od nule korišteni su okidači. Njima baza podataka reagira na unos, izmjenu ili brisanje n-torke iz neke relacije tako da provjerava uvjet te ako je zadovoljen, izvodi akciju ili poziva pohranjenu proceduru. Stanje skladišta smanjuje se unosom stavki računa, odnosno izdavanjem robe, a povećava zaprimanjem. Pohranjene procedure i okidači za dodavanje, brisanje i izmjenu stavki računa navedeni su u nastavku (Kôd 2.2, Kôd 2.3, Kôd 2.4), a analogno se definiraju za stavke primki, uz zamjenu smanjenja stanja za povećanje. Ukoliko pohranjena procedura vrati NULL, dogodit će se poništavanje dotad obavljenih operacija.

```
CREATE OR REPLACE FUNCTION sync_stanje_insert() RETURNS
trigger AS
$$
BEGIN
    UPDATE ima_na_stanju
    SET kolicinastanje = kolicinastanje -
    NEW.kolicinaproizvodracun
    FROM racun
    WHERE sifproizvod = NEW.sifproizvod
        AND racun.sifracun = NEW.sifracun
        AND ima_na_stanju.sifskladiste =
        racun.sifskladiste;

    IF ((SELECT kolicinastanje
        FROM ima_na_stanju
        JOIN racun ON
        ima_na_stanju.sifskladiste = racun.sifskladiste
        WHERE sifproizvod = NEW.sifproizvod
            AND sifracun = NEW.sifracun)
        < 0) THEN
        RAISE EXCEPTION
        'Error: stanje proizvoda ne može biti negativno';
```

```

        RETURN NULL;
    ELSE
        RETURN NEW;
    END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER ins_sync_stanje
    BEFORE INSERT ON stavka_racuna
    FOR EACH ROW
    WHEN (NEW.kolicinaproizvodracun <> 0)
    EXECUTE PROCEDURE sync_stanje_insert();

```

Kôd 2.2 Pohranjena procedura i okidač za INSERT u Stavka_Racun

```

CREATE OR REPLACE FUNCTION sync_stanje_delete() RETURNS
trigger AS
$$
    BEGIN
        UPDATE ima_na_stanju
        SET kolicinastanje =
            kolicinastanje + OLD.kolicinaproizvodracun
        FROM racun
        WHERE sifproizvod = OLD.sifproizvod
            AND racun.sifracun = OLD.sifracun
            AND ima_na_stanju.sifskladiste =
                racun.sifskladiste;
        RETURN OLD;
    END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER del_sync_stanje
    BEFORE DELETE ON stavka_racuna
    FOR EACH ROW
    WHEN (OLD.kolicinaproizvodracun <> 0)
    EXECUTE PROCEDURE sync_stanje_delete();

```

Kôd 2.3 Pohranjena procedura i okidač za DELETE na Stavka_racun

```

CREATE OR REPLACE FUNCTION sync_stanje_update() RETURNS
trigger AS
$$
    BEGIN

```



```

UPDATE ima_na_stanju
SET kolicinastanje = kolicinastanje -
(NEW.kolicinaproizvodracun - OLD.kolicinaproizvodracun)
FROM racun
WHERE ima_na_stanju.sifproizvod = OLD.sifproizvod
      AND racun.sifracun = OLD.sifracun
      AND ima_na_stanju.sifskladiste =
      racun.sifskladiste;

IF ((SELECT kolicinastanje
      FROM ima_na_stanju
      JOIN racun ON
      ima_na_stanju.sifskladiste = racun.sifskladiste
      WHERE sifproizvod = NEW.sifproizvod
            AND sifracun = NEW.sifracun)
    < 0) THEN
    RAISE EXCEPTION
    'Error: stanje proizvoda ne moze biti negativno';
    RETURN NULL;
ELSE
    RETURN NEW;
END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER upd_sync_stanje
BEFORE UPDATE OF kolicinaproizvodracun ON stavka_racuna
FOR EACH ROW
WHEN (OLD.kolicinaproizvodracun <>
      NEW.kolicinaproizvodracun)
EXECUTE PROCEDURE sync_stanje_update();

```

Kôd 2.4 Pohranjena procedura i okidač za UPDATE na Stavka_Racun

3. Web aplikacija

3.1. Korištene tehnologije

Git

Sustav za upravljanje izvornim kodom baziran na repozitorijima. Olakšava praćenje promjena, vraćanje na prijašnje stanju u kodu, pruža funkcionalnosti poput stvaranja i spajanja grana. Za udaljeni repozitorij projekta koristila se web platforma GitLab.

Spring Boot

Spring je široki skup biblioteka i alata za razvoj aplikacija u Javi, najčešće se koristi putem anotacija, a bazira se na oblikovnim obrascima poput predloška i aspektno orijentiranog programiranja. Zasniva se na aplikacijskom kontekstu kojim izvršava injekciju ovisnosti. Spring Boot je modul Spring radnog okvira koji se primarno koristi za razvoj REST API-ja i glavna značajka mu je automatska konfiguracija te dolazi s ugrađenim Tomcat serverom. U radu je korišten za razvoj backenda.

Eclipse

Integrirana razvojna okolina za Java programski jezik s mnoštvom dostupnih modula, primjerice ovdje korišten Spring Tools Suite.

React.js

React.js je najpoznatija JavaScript biblioteka otvorenog koda namijenjena razvoju interaktivnih korisničkih sučelja. Zasniva se na komponentama, upravljanjem stanja tih komponenti i njihovom prikazivanju.

Visual Studio Code

Uređivač izvornog koda s brojim korisnim funkcionalnostima poput inteligentnog predviđanja koda, naglašavanja sintakse i refaktoriranja koda. Primarno je namijenjen izradi modernih web aplikacija. U radu je korišten za razvoj frontenda zajedno s React.js bibliotekom.

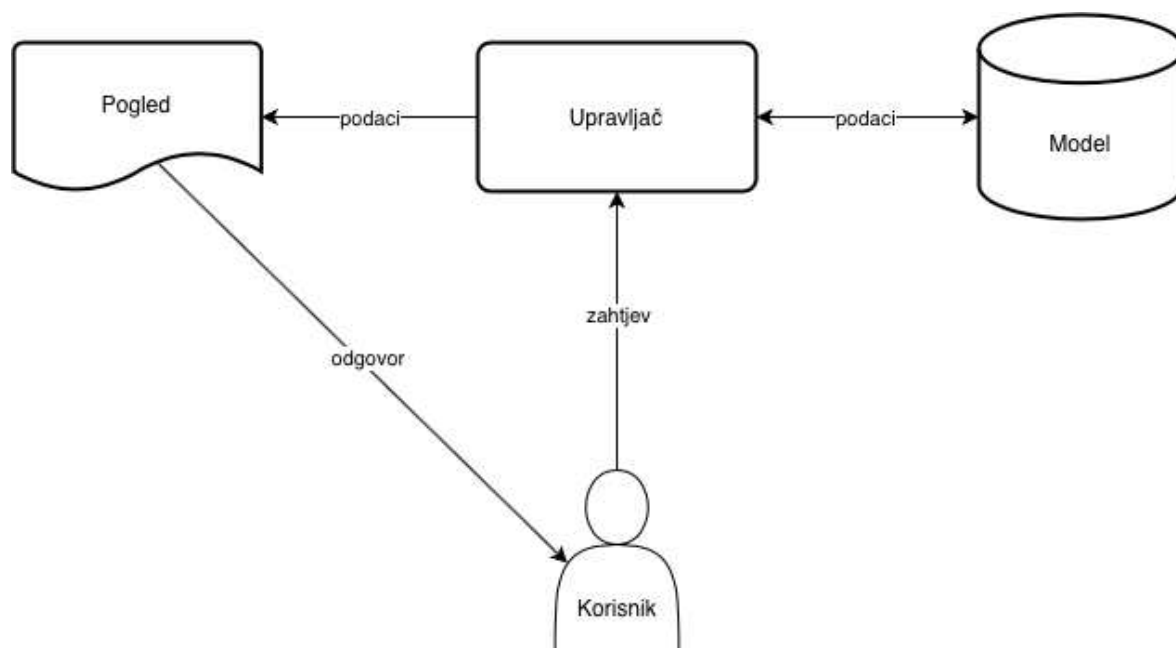
Postman

Alat za slanje http upita na server s opširnim opcijama postavljanja tijela zahtjeva, autorizacijskih tokena te prikaza dobivenog odgovora.

3.2. Arhitektura

Odabir načina oblikovanja arhitekture jedna je od najvažnijih odluka pri modeliranju web aplikacije jer uvjetuje načine oblikovanja, cijenu i težinu implementacije, lakoću nadogradnje te brojne druge faktore uspješnosti rada.

Odabrana je arhitektura Model-Pogled-Upravljač, odnosno MVC (engl. *Model-View-Controller*), a generalizacija te arhitekture prikazana je na slici (Sl. 3.1).



Sl. 3.1 Prikaz MVC arhitekture

Praćenje MVC obrasca pospješuje inverziju i razdvajanje ovisnosti te tako olakšava istovremeni razvoj više dijelova aplikacije, nadogradnju funkcionalnosti i testiranje. Primjer toga je razvoj i testiranje cijelih Model i Upravljač slojeva pomoću alata Postman, prije nego je izrađen Pogled sloj arhitekture.

3.3. Implementacija

Nakon završene baze podataka, slijedio je postupak implementacije web aplikacije koji se sastojao od dvije faze, razvoja backenda te razvoja frontenda.

3.3.1. Backend

Backend izveden pomoću Spring Boota sastoji se od tri sloja. Prvi sloj za pristup podacima obuhvaćao je modele i repozitorije. Model je klasa koja predstavlja tablicu u bazi podataka

i, uz pomoć Spring anotacija, omogućava automatsko preslikavanje. Dodatno, umjesto stranih ključeva, modeli sadrže reference na druge modele, što je vidljivo u kodu (Kôd 3.1).

```
@Entity
@Table(name= "proizvod")
@JsonIgnoreProperties({"imaNaStanjus", "stavkeRacuna",
"stavkePrimke", "rabats"})
public class Proizvod {

    @Id
    @GeneratedValue
    @Column(name = "sifproizvod")
    private int sifProizvod;

    @Column(name = "barkod")
    private String barkodProizvod;

    @Column(name = "nazivproizvod")
    private String nazivProizvod;

    @Column(name = "mjera")
    private String mjeraProizvod;

    @Column(name = "vrsta")
    private String vrstaProizvod;

    @Column(name = "podgrupa")
    private String podgrupaProizvod;

    @Column(name = "kolicinaukutiji")
    private Integer kolicinaKutijaProizvod;

    @Column(name = "masa")
    private Double masaProizvod;

    @Column(name = "cijena")
    private Double cijenaProizvod;

    @ManyToOne
    @JoinColumn(name = "sifproizvodac")
```

```

private Proizvodac proizvodac;

@OneToMany(mappedBy = "proizvod")
private Set<ImaNaStanju> imaNaStanjus;

@OneToMany(mappedBy = "proizvod")
private Set<StavkaRacuna> stavkeRacuna;

@OneToMany(mappedBy = "proizvod")
private Set<StavkaPrimke> stavkePrimke;

@OneToMany(mappedBy = "proizvod")
private Set<Rabat> rabats;

```

Kôd 3.1 Primjer modela za tablicu Proizvod

Repozitorij je naziv za komponentu kojom se dohvaćaju podaci iz baze podataka i preslikavaju u model, a primjer je prikazan kodom (Kôd 3.2).

```

@Transactional
public interface ProizvodRepository extends
JpaRepository<Proizvod, Integer> {
    Optional<Proizvod> findByNazivProizvod(String naziv);
    Boolean existsByNazivProizvod(String naziv);
}

```

Kôd 3.2 Primjer repozitorija za proizvode

Nakon izrađenog sloja za pristup podacima, oblikovan je drugi, servisni sloj. Za svaki model definirano je općenito sučelje s deklaracijom metoda koje servis treba omogućavati (Kôd 3.3), te konkretna implementacija tog sučelja u kojoj se vidi primjena Spring anotacije `@Autowired` koja izvršava automatsku konfiguraciju odnosa između komponenti (Kôd 3.4).

```

public interface ProizvodService {
    List<Proizvod> listAllProizvods();
    Optional<Proizvod> findByNaziv(String naziv);
    Boolean existsByNaziv(String naziv);
    Optional<Proizvod> findBySifProizvod(Integer sif);
    Proizvod createProizvod(Proizvod proizvod);
    void deleteProizvod(Integer sif);
}

```

Kôd 3.3 Sučelje servisa za proizvode

```

@Service
public class ProizvodServiceJpa implements ProizvodService {

    @Autowired
    ProizvodRepository proizvodRepo;

    @Override
    public List<Proizvod> listAll() {
        return proizvodRepo.findAll();
    }

    @Override
    public Optional<Proizvod> findByNaziv(String naziv) {
        return proizvodRepo.findByNazivProizvod(naziv);
    }

    @Override
    public Boolean existsByNaziv(String naziv) {
        return proizvodRepo.existsByNazivProizvod(naziv);
    }

    @Override
    public Optional<Proizvod> findBySifProizvod(Integer
sif) {
        return proizvodRepo.findById(sif);
    }

    @Override
    public Proizvod createProizvod(Proizvod proizvod) {
        return proizvodRepo.save(proizvod);
    }

    @Override
    public void deleteProizvod(Integer sif) {
        proizvodRepo.deleteById(sif);
        return;
    }

}

```

Kôd 3.4 Implementacija sučelja servisa za proizvode

Treći sloj je upravljački. Pojedine upravljačke klase prihvataju http zahtjeve na adresama za koje su definirane te određuju i generiraju odgovor. Primjer upravljačke klase i metode za prihvatanje GET zahtjeva za sve proizvode i pojedinačan proizvod prikazan je kodom (Kôd 3.5).

```

@CrossOrigin(origins = "*", maxAge = 3600)

```

```

@RestController
@RequestMapping("/api/proizvod")
public class ProizvodController {

    private static final Logger logger =
        LoggerFactory.getLogger(ProizvodController.class);

    @Autowired
    private ProizvodService proizvodService;
    @Autowired
    private ProizvodacService proizvodacService;

    @GetMapping("/all")
    @PreAuthorize("hasRole('ROLE_ADMIN') or
        hasRole('ROLE_DIREKTOR') or
        hasRole('ROLE_SKLADISTAR')")
    public ResponseEntity<List<Proizvod>> getProizvods() {
        return
            ResponseEntity.ok(proizvodService.listAll());
    }

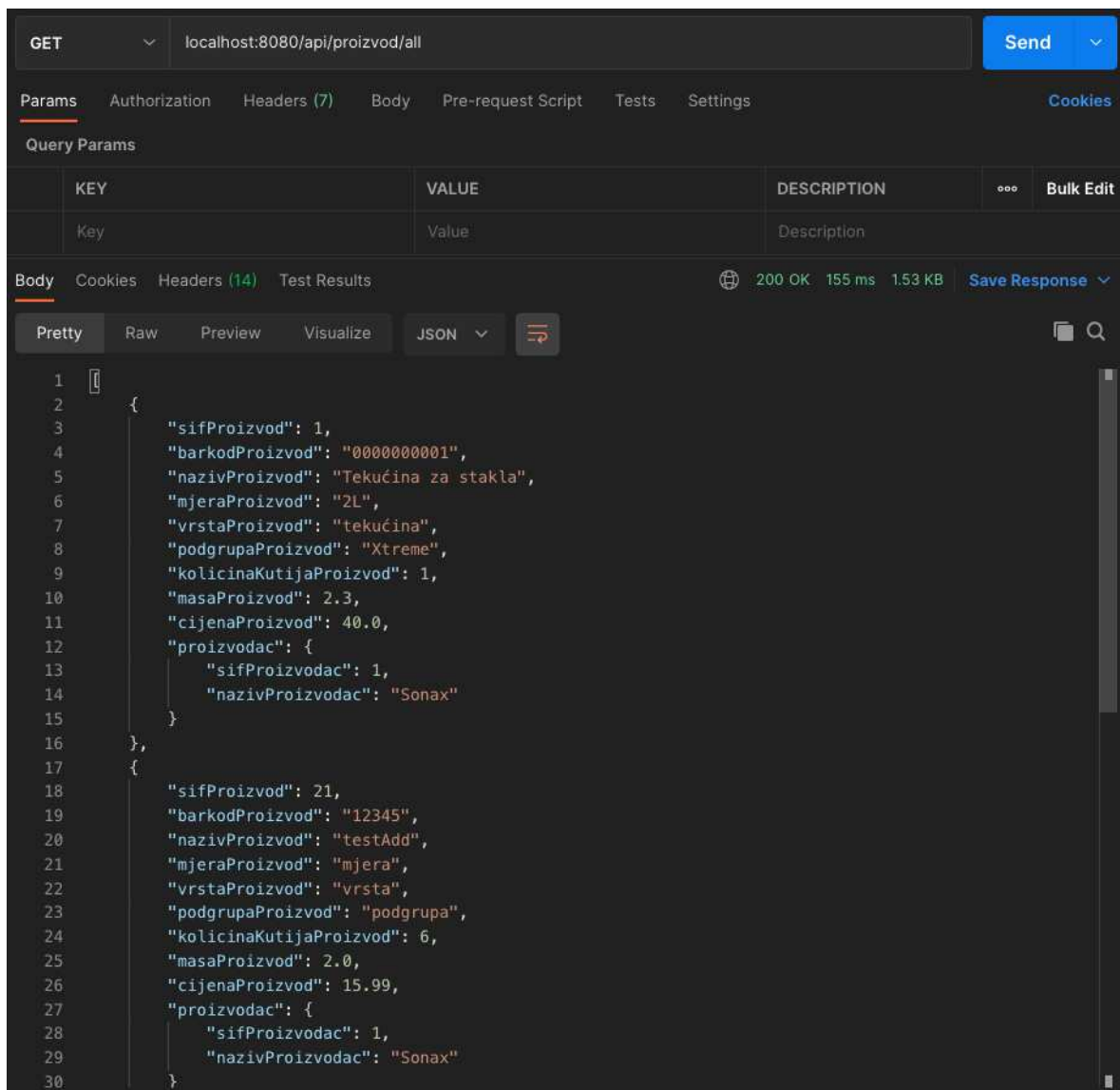
    @GetMapping("/{id}")
    @PreAuthorize("hasRole('ROLE_ADMIN') or
        hasRole('ROLE_DIREKTOR') or
        hasRole('ROLE_SKLADISTAR')")
    public ResponseEntity<Proizvod>
        getProizvod(@PathVariable Integer id) {
        Optional<Proizvod> proizvod =
            proizvodService.findBySifProizvod(id);

        if(proizvod.isEmpty()) {
            logger.error("Proizvod id: " + id + " does
                not exist");
            return ResponseEntity.badRequest().build();
        }
        return ResponseEntity.ok(proizvod.get());
    }
}

```

Kôd 3.5 Dio upravljača za proizvode

Po završetku sva tri sloja obavljeno je testiranje funkcionalnosti pomoću alata Postman, a primjer je prikazan slikom (Sl. 3.2).



Sl. 3.2 Primjer testiranja backenda alatom Postman

3.3.2. Frontend

U drugoj fazi razvoja implementiran je frontend uz pomoć React.js radnog okvira. Za svaku komponentu prvo je definirana servisna klasa koja obavlja slanje zahtjeva na backend (Kôd 3.6).

```
class ProizvodService {
  async allProizvods() {
    try {
      let response = await axios.get(`/proizvod/all`)
      console.log(response.data)
      return response.data
    } catch (err) {
      console.log(err)
    }
  }
}
```



```

        return null
    }
}
async getProizvod(id) {
    try {
        let response = await axios.get(`/proizvod/${id}`)
        console.log(response.data)
        return response.data
    } catch(err) {
        console.log(err)
        return null
    }
}
async updateProizvod(id, payload) {
    try {
        let response = await axios.put(`/proizvod/${id}`,
        payload)
        console.log(response.data)
        return response
    } catch(err) {
        console.log(err)
        return null
    }
}
async createProizvod(payload) {
    try {
        let response = await axios.post(`/proizvod`,
        payload)
        console.log(response.data)
        return response
    } catch(err) {
        console.log(err)
        return null
    }
}
async deleteProizvod(id) {
    try {
        let response = await
        axios.delete(`/proizvod/${id}`)
        console.log(response.data)
        return response
    }
}

```

```

        } catch(err) {
            console.log(err)
            return null
        }
    }
}

```

Kôd 3.6 Klasa s metodama za slanje zahtjeva na backend

Metode klase opisane iznad koriste se u komponentama za postavljanje stanja te spremanje izmijenjenih ili novih podataka. Primjer komponente prikazan je kodom (Kôd 3.7), uz napomenu da je metoda 'render' za prikaz komponente izostavljena zbog duljine i preglednosti.

```

class ListProizvodsComponent extends Component {

    constructor(props) {
        super(props)
        this.state = {
            proizvodi: [],
            role: null
        }

        this.deleteProizvodClicked =
        this.deleteProizvodClicked.bind(this)
        this.updateProizvodClicked =
        this.updateProizvodClicked.bind(this)
        this.addProizvodClicked =
        this.addProizvodClicked.bind(this)
        this.refreshProizvodi =
        this.refreshProizvodi.bind(this)
    }

    async componentDidMount() {
        let resMe = await UserDataService.currentUser()
        this.setState({
            role: resMe.radnoMj.nazivRadnoMjesto
        })
        this.refreshProizvodi();
    }
}

```

```

async refreshProizvodi() {
    let response = await ProizvodService.allProizvods();
    this.setState({
        proizvodi: response
    })
}

async deleteProizvodClicked(id) {
    let response = await
    ProizvodService.deleteProizvod(id)
    console.log('deleteProizvodClicked got status: ' +
    response.status)
    if (response.status === 200) {
        this.refreshProizvodi()
    }
}

updateProizvodClicked(id) {
    console.log('update ' + id)
    this.props.history.push(`/proizvod/${id}`)
}

addProizvodClicked() {
    console.log('add proizvod clicked')
    this.props.history.push(`/addProizvod`)
}

render() {
    ...
}

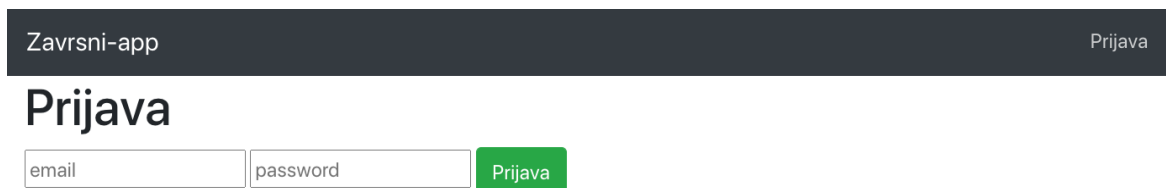
```

Kôd 3.7 Komponenta za prikaz svih proizvoda

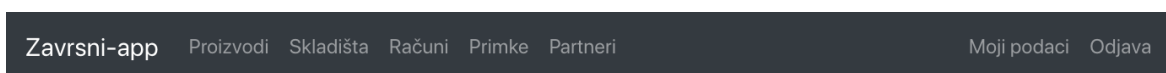
Za sigurnost web aplikacije korišten je JWT (engl. *JSON Web Token*), industrijski standard otvorenog koda za osiguravanje komunikacije preko mreže. Velika prednost nad običnom autorizacijom koju nudi Spring Security je da se može namjestiti vremenski istek valjanosti tokena. Generiranje kriptografskog sažetka lozinke obavlja se algoritmom BCrypt.

3.4. Upute za korištenje

Nakon pokretanja aplikacije prvo se potrebno prijaviti (Sl. 3.3). Korisnici se ne registriraju sami, već sve korisnike, odnosno zaposlenike u sustav unosi administrator, prilikom njihovog zaposlenja. Po uspješnoj prijavi, prikazuje se početna stranica s dobrodošlicom (Sl. 3.4), a daljnje funkcionalnosti dostupne su putem navigacijske trake i određene su prema radnom mjestu.



Sl. 3.3 Stranica za prijavu



Welcome, Ivan Horvat

Sl. 3.4 Početna stranica

3.4.1. Zaposlenik u skladištu

Od svih predviđenih korisnika, ima najmanje ovlasti. Klikom na 'Moji podaci' u navigacijskoj traci može pregledati svoje podatke (Sl. 3.5), ali ako uoči pogrešku mora to prijaviti administratoru kako bi se ispravila.

| | |
|---------------------|---------------------|
| OIB | 55555555555 |
| Ime | Skladiško |
| Prezime | Skladiš |
| Email | skladistar@email.hr |
| Broj telefona | 3092535 |
| Datum rođenja | 1988-12-02 |
| Datum početka rada | 2007-06-08 |
| Datum kraja rada | |
| Naziv radnog mjesta | skladistar |
| Grad | Zagreb |
| Adresa | Zagrebačka 2 |

Sl. 3.5 Stranica za pregled vlastitih podataka

Na poveznici 'Proizvodi' može pregledavati sve proizvode u asortimanu tvrtke (Sl. 3.6).

Svi proizvodi

| Šifra | Naziv | Mjera | Količina/kutija | Masa | Cijena |
|-------|----------------------|-------|-----------------|------|--------|
| 1 | Tekućina za stakla | 2L | 1 | 2.3 | 40 |
| 21 | testAdd | mjera | 6 | 2 | 15.99 |
| 22 | Wunder-Baum Vanilija | 1kom | 12 | 0.1 | 7 |
| 24 | testAdd2 | komad | 6 | 6 | 6 |

Sl. 3.6 Stranica za pregled svih proizvoda

Klikom na 'Skladišta' može pregledati sva skladišta (Sl. 3.7)., a klikom na 'Stanje' pojedinog skladišta može vidjeti stanje svih proizvoda na tom skladištu (Sl. 3.8).

| Zavrsni-app | Proizvodi | Skladišta | Računi | Primke | Partneri | Moji podaci | Odjava |
|---------------|------------|--------------------|--------|--------|----------|-------------|--------|
| Sva skladišta | | | | | | | |
| Šifra | Naziv | Adresa | Stanje | | | | |
| 1 | skladiste1 | adresa skladiste 1 | Stanje | | | | |
| 2 | skladiste2 | adresa skladiste 2 | Stanje | | | | |

Sl. 3.7 Stranica za pregled svih skladišta

| Zavrsni-app | Proizvodi | Skladišta | Računi | Primke | Partneri | Moji podaci | Odjava |
|-----------------------|----------------------|-----------|--------|--------|----------|-------------|--------|
| Skladište: skladiste1 | | | | | | | |
| Šifra | Naziv | Stanje | | | | | |
| 24 | testAdd2 | 85499 | | | | | |
| 1 | Tekućina za stakla | 88490 | | | | | |
| 21 | testAdd | 85500 | | | | | |
| 22 | Wunder-Baum Vanilija | 86600 | | | | | |

Sl. 3.8 Stranica za pregled stanja proizvoda na skladištu

Poveznicom 'Računi' u navigacijskoj traci dobiva listu svih računa (Sl. 3.9), gdje može brisati i dodavati račune. Prilikom dodavanja računa, nakon klika na gumb 'Add', u informacije računa je već učitano kao kreator računa i to ne može promijeniti, odnosno ne može unijeti račun u tuđe ime, a vrijednosti stranih ključeva sifSkladiste i sifPoslovnica imaju padajući izbornik za izbor vrijednosti (Sl. 3.10)

Završni-app

Proizvodi

Skladišta

Računi

Primke

Partneri

Moji podaci

Odjava

Svi računi

| Šifra | Datum | Valuta | Ukupna cijena | Plaćen | Detalji | |
|-------|------------|--------|---------------|--------|-------------------------|------------------------|
| 55 | 2020-08-07 | kn | 125986 | da | Detalji | Delete |
| 26 | 2021-05-05 | eur | 0 | ne | Detalji | Delete |
| 1 | 2018-12-12 | kn | 37385 | ne | Detalji | Delete |
| 34 | 2021-05-31 | kn | 35400 | ne | Detalji | Delete |
| 25 | 2021-05-04 | kn | 4700 | ne | Detalji | Delete |
| 54 | 2020-03-07 | kn | 7995 | da | Detalji | Delete |

Add

Sl. 3.9 Stranica s prikazom svih računa

| | | | | | | | | | | | | | | |
|--|---|--|--|--|--|--|--|--|--|--|--|--|--|--|
| Zavrsni-app Proizvodi Skladišta Računi Primke Partneri | | | | | | Moji podaci Odjava | | | | | | | | |
| Račun | | | | | | | | | | | | | | |
| Datum: | <input type="text" value="mm/dd/yyyy"/> | | | | | | | | | | | | | |
| Valuta: | <input type="text"/> | | | | | | | | | | | | | |
| Plaćen: | <input type="text"/> | | | | | | | | | | | | | |
| Skladište: | <div> <input checked="" type="checkbox"/> skladiste1 <input type="checkbox"/> skladiste2 </div> | | | | | | | | | | | | | |
| Poslovnica | <input type="text"/> | | | | | | | | | | | | | |
| Šifra zaposlenika: | <input type="text" value="57"/> | | | | | | | | | | | | | |
| <input type="button" value="Spremi"/> | | | | | | | | | | | | | | |

Sl. 3.10 Stranica za unos novog računa

Nakon uspješnog dodavanja računa, ili odabirom gumba 'Detalji' pojedinog računa iz liste, otvara se stranica s detaljima tog računa i listom stavki. Također se može brisati postojeće i dodavati nove stavke računa odabirom proizvoda iz padajućeg izbornika te unosom količine tog proizvoda. Ukupna cijena računa automatski se izračunava i ažurira (Sl. 3.11).

Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Moji podaci Odjava

Račun

| | |
|---------------|---------------------|
| Šifra računa | 59 |
| Datum | 2021-06-09 |
| Plaćen | ne |
| Skladište | skladiste1 |
| Izdao | skladistar@email.hr |
| Poslovnica | Petrol Branimirova |
| Ukupna cijena | 4000 |
| Valuta | kn |

Stavke

| Proizvod | Količina | |
|--------------------|----------|--------|
| Tekućina za stakla | 100 | Delete |

✓

Tekućina za stakla

testAdd

Wunder-Baum Vanilija

testAdd2

Dodaj

Sl. 3.11 Stranica za pregled detalja računa te brisanje i dodavanje stavki

Analogne funkcionalnosti računu, implementirane su i za primke. Klikom na 'Primke' u navigacijskoj traci otvara se lista svih primki, gdje onda može brisati postojeće i dodavati nove primke, vidjeti njihove detalje te brisati i dodavati zaprimljene proizvode unoseći njihovu količinu i nabavnu cijenu (Sl. 3.12, Sl. 3.13, Sl. 3.14).

Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Moji podaci Odjava

Sve primke

| Šifra | Datum | Valuta | Detalji |
|-------|------------|--------|--------------------------------------|
| 41 | 2021-06-05 | kn | <div>Detalji</div> <div>Delete</div> |

Add

Sl. 3.12 Stranica s pregledom svih primki

Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Moji podaci Odjava

localhost:3000 says
Uspjesno dodana primka
OK

Primka

Datum:

06/10/2021

Valuta:

kn

Skladište:

skladiste1

Partner:

Sonax

Spremi

Sl. 3.13 Stranica za dodavanje primke

Primka

| | |
|------------------|------------|
| Šifra primke | 61 |
| Datum | 2021-06-10 |
| Skladište | skladiste1 |
| Poslovni partner | Sonax |
| Valuta | kn |

Stavke

| Proizvod | Količina | Kupovna cijena | |
|----------------------|----------------------|----------------------|--------|
| Tekućina za stakla | 200 | 28 | Delete |
| <input type="text"/> | <input type="text"/> | <input type="text"/> | Dodaj |

Sl. 3.14 Stranica za pregled detalja primke, brisanje i dodavanje zaprimljenih proizvoda

Zadnje funkcionalnosti predviđene za radnika u skladištu jesu pregled svih poslovnih partnera dostupan klikom na 'Partneri' (Sl. 3.15), gdje onda može vidjeti rabate za svaki proizvod ugovorene s tim poslovnim partnerom klikom na gumb 'Rabati' (Sl. 3.16) te pregled svih poslovnica tog poslovnog partnera klikom na gumb 'Poslovnice' (Sl. 3.17).

Svi poslovni partneri

| Šifra | Naziv | Adresa | | |
|-------|----------|---------------------|--------|------------|
| 1 | Petrol | Petrolska ulica 1 | Rabati | Poslovnice |
| 27 | Sonax | adresa u Njemačkoj | Rabati | Poslovnice |
| 32 | Crodux | Savska Opatovina 36 | Rabati | Poslovnice |
| 44 | Kaufland | Donje Svetice 14 | Rabati | Poslovnice |
| 56 | Plodine | Plodinska cesta 3 | Rabati | Poslovnice |

Sl. 3.15 Stranica s popisom svih partnera

Poslovni partner: Crodex

| Šifra proizvod | Naziv | Iznos rabat(%) |
|----------------|----------------------|----------------|
| 21 | testAdd | 25 |
| 1 | Tekućina za stakla | 8 |
| 22 | Wunder-Baum Vanilija | 20 |
| 24 | testAdd2 | 19 |

Sl. 3.16 Stranica za pregled ugovorenih rabata za proizvode

Poslovni partner: Petrol

| Šifra | Naziv | Broj tel. | Adresa |
|-------|--------------------|-----------|------------------------------|
| 1 | Petrol Branimirova | 01392775 | Branimirova 121, Zagreb |
| 31 | testAdd | 23445345 | adresaPoslovniceTest, Zagreb |

Sl. 3.17 Stranica za pregled svih poslovnica određenog partnera

3.4.2. Direktor tvrtke

Uključujući sve navedene funkcionalnosti koje ima radnik u skladištu, direktor ima dodatne ovlasti. Prvo proširenje ovlasti je mogućnost dodavanja, izmjene i brisanja proizvoda kada se nalazi na stranici za pregled svih proizvoda (Sl. 3.18, Sl. 3.19).

Svi proizvodi

| Šifra | Naziv | Mjera | Količina/kutija | Masa | Cijena | | |
|-------|----------------------|-------|-----------------|------|--------|------|--------|
| 1 | Tekućina za stakla | 2L | 1 | 2.3 | 40 | Edit | Delete |
| 21 | testAdd | mjera | 6 | 2 | 15.99 | Edit | Delete |
| 22 | Wunder-Baum Vanilija | 1kom | 12 | 0.1 | 7 | Edit | Delete |
| 24 | testAdd2 | komad | 6 | 6 | 6 | Edit | Delete |

Add

Sl. 3.18 Stranica za pregled svih proizvoda za direktora

Proizvod

Barkod:

Naziv:

Mjera:

Vrsta:

Podgrupa:

Količina u kutiji:

Masa:

Cijena:

Proizvođač:

✓ Sonax
 Wunder-Baum
 MyShaldan

Spremi

Sl. 3.19 Stranica za uređivanje/dodavanje proizvoda

Također, može dodavati nova skladišta pritiskom na gumb 'Add' na stranici s listom svih skladišta (Sl. 3.20).

Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Zaposlenici Statistika Moji podaci Odjava

Skladište

Naziv skladište:

Površina (m2):

Adresa skladišta:

Grad:

Spremi

Sl. 3.20 Stranica za dodavanje novog skladišta

Sljedeće proširenje je da prilikom prikaza liste svih poslovnih partnera ima opcije dodavanja, brisanja i uređivanja poslovnih partnera (Sl. 3.21), a na listi svih poslovnica određenog partnera može dodavati i brisati poslovnice (Sl. 3.22).

Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Zaposlenici Statistika Moji podaci Odjava

Svi poslovni partneri

| Šifra | Naziv | Adresa | | | | |
|-------|----------|---------------------|--------|------------|------|--------|
| 1 | Petrol | Petrolska ulica 1 | Rabati | Poslovnice | Edit | Delete |
| 27 | Sonax | adresa u Njemačkoj | Rabati | Poslovnice | Edit | Delete |
| 32 | Crodux | Savska Opatovina 36 | Rabati | Poslovnice | Edit | Delete |
| 44 | Kaufland | Donje Svetice 14 | Rabati | Poslovnice | Edit | Delete |
| 56 | Plodine | Plodinska cesta 3 | Rabati | Poslovnice | Edit | Delete |

Add

Sl. 3.21 Stranica s listom svih poslovnih partnera za direktora

Poslovni partner: Petrol

| Šifra | Naziv | Broj tel. | Adresa | |
|-------|--------------------|-----------|------------------------------|--------|
| 1 | Petrol Branimirova | 01392775 | Branimirova 121, Zagreb | Delete |
| 31 | testAdd | 23445345 | adresaPoslovniceTest, Zagreb | Delete |

Add

Sl. 3.22 Stranica s mogućnosti dodavanja i brisanja poslovnica poslovnog partnera

Dodatna funkcionalnost koju radnik u skladištu nije imao je pregled svih zaposlenika tvrtke vidljiv na slici (Sl. 3.23).

Svi zaposlenici

| Šifra | Ime | Prezime | Email |
|-------|-----------|---------|---------------------|
| 9 | adminko | admin | admin@email.hr |
| 1 | Ivan | Horvat | ihorvat@email.hr |
| 57 | Skladiško | Skladiš | skladistar@email.hr |

Sl. 3.23 Stranica s pregledom svih zaposlenika

Među bitnijim funkcionalnostima za direktora valja istaknuti stranicu dostupnu klikom na 'Statistika' u navigacijskoj traci (Sl. 3.24). Na toj stranici nalaze se bitne opcije za analizu prodaje i uspješnosti tvrtke.

Završni-app Proizvodi Skladišta Računi Primke Partneri Zaposlenici Statistika Moji podaci Odjava

Godina

2018

2019

2020

2021

Prodaja po kupcima

- Petrol
- Sonax
- Crodex
- Kaufland
- Plodine

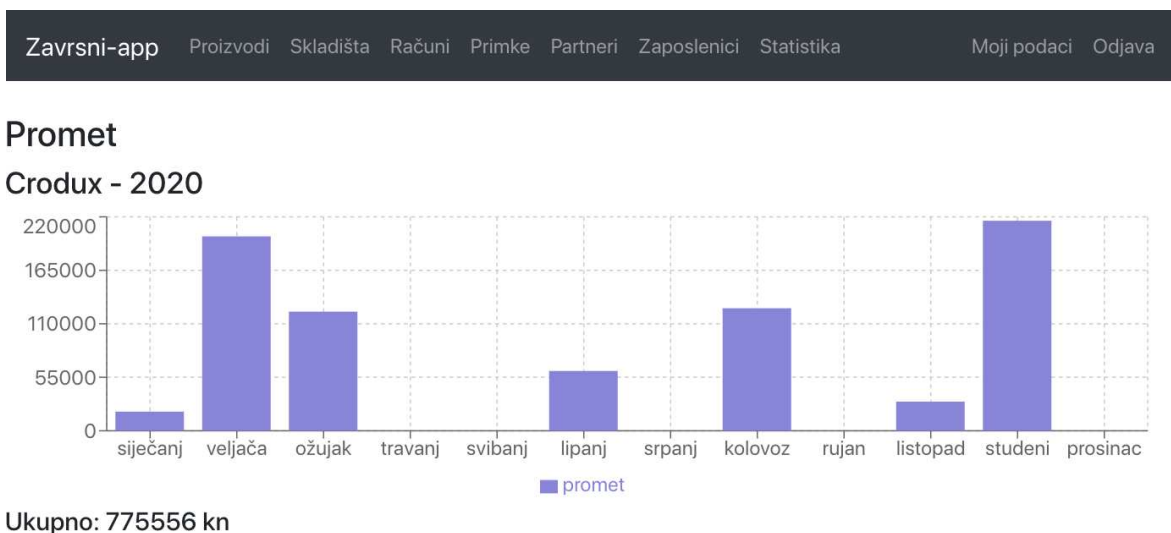
Prodaja po proizvodima

- Tekućina za stakla
- testAdd
- Wunder-Baum Vanilija
- testAdd2

Usporedba proizvoda

Sl. 3.24 Stranica s dostupnim raznim statistikama

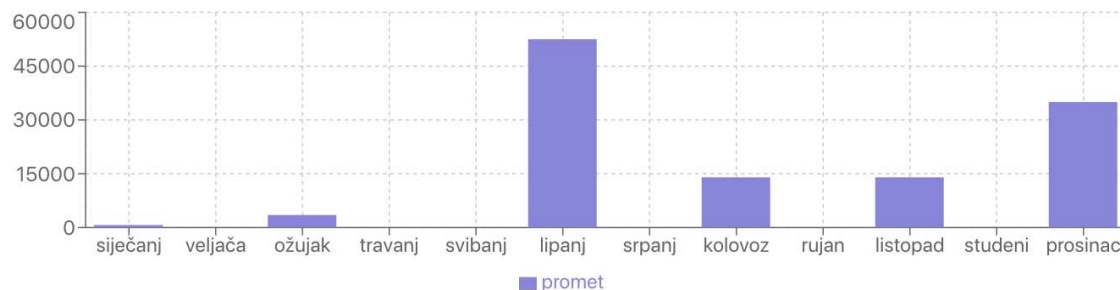
Nakon odabira godine za koju se želi prikazati statistika, može se odabrati prikaz analize prodaje po mjesecima za određenog poslovnog partnera (Sl. 3.25) ili određeni proizvod (Sl. 3.26).



Sl. 3.25 Stranica s prikazom prometa za partnera po mjesecima u 2020. godini

Promet

Wunder-Baum Vanilija - 2020

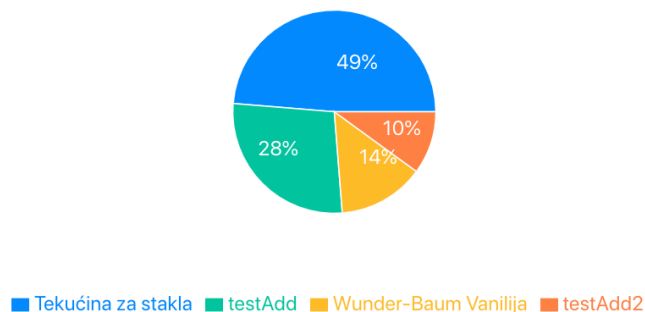


Ukupno: 119700 kn

Sl. 3.26 Stranica s prikazom prometa za proizvod po mjesecima u 2020. godini

Zadnja dostupna statistika je usporedba proizvoda po prodajnoj vrijednosti u odabranoj godini koja se prikazuje na sljedeći način (Sl. 3.27).

Usporedba prometa po proizvodima - 2020



Sl. 3.27 Stranica s prikazom usporedbe prometa proizvoda u odabranoj godini

3.4.3. Administrator

Ulogu administratora ima razvojni tim koji je zadužen za razvoj i održavanje web aplikacije. Administrator ima apsolutne ovlasti unutar aplikacije, što znači da pored svih nabrojanih ovlasti radnika u skladištu i direktora tvrtke ima mogućnost izmjene, brisanja i dodavanja novih zaposlenika. Prikaz svih zaposlenika za administratora ima navedene

opcije (Sl. 3.28), dok je primjer uređivanja podataka zaposlenika prikazan na slici (Sl. 3.29).

Zavrsni-app

Proizvodi

Skladišta

Računi

Primke

Partneri

Zaposlenici

Statistika

Moji podaci

Odjava

Svi zaposlenici

| Šifra | Ime | Prezime | Email | | |
|-------|-----------|---------|---------------------|------|--------|
| 9 | adminko | admin | admin@email.hr | Edit | Delete |
| 1 | Ivan | Horvat | ihorvat@email.hr | Edit | Delete |
| 57 | Skladiško | Skladiš | skladistar@email.hr | Edit | Delete |

Add

Sl. 3.28 Stranica s listom svih zaposlenika za administratora

| | |
|---|------------------|
| Zavrsni-app Proizvodi Skladišta Računi Primke Partneri Zaposlenici Statistika Moji podaci Odjava | |
| Zaposlenik | |
| OIB: | 12345678901 |
| Ime: | Ivan |
| Prezime: | Horvat |
| Email: | ihorvat@email.hr |
| Tel: | 099123456789 |
| Datum rođenja: | 09/24/1973 |
| Datum početka rada: | 09/24/2001 |
| Datum kraja rada: | mm/dd/yyyy |
| Novi password: | |
| Radno mjesto | direktor |
| Grad | Zagreb |
| Adresa: | Adresna Ulica 18 |
| Spremi | |

Sl. 3.29 Stranica za uređivanje/dodavanje podataka zaposlenika

Zaključak

Cilj ovog rada bio je modelirati i implementirati funkcionalnu bazu podataka i web aplikaciju koja bi se mogla koristiti u veleprodajnoj tvrtki. Ideja za rad primarno je došla iz osobnog radnog iskustva što je uvelike pomoglo izdvajanju bitnih funkcionalnosti i motivaciji za uspješno izvršavanje zadatka.

Proces izgradnje navedenog sustava opisan je u ovom radu, a najveći izazov predstavljala je izrada korisničkog sučelja tehnologijom React.js zbog nedostatka prethodnog iskustva, dok su implementiranje baze i upravljačkog sloja u PostgreSQL-u i Spring Boot-u prošli bez poteškoća zbog rada s navedenim tehnologijama na prijašnjim projektima.

Primarni cilj daljnjih poboljšanja ovog rada bila bi vizualna nadogradnja korisničkog sučelja te olakšavanje unosa podataka, primarno uvoz i izvoz podataka iz .xml datoteka koje su popularne u trgovini i često se šalju kao privitak računu. Još jedna korisna funkcionalnost bila bi automatsko generiranje PDF dokumenta za račune sa svim popratnim stavkama i informacijama.

Literatura

- [1] Nastavni materijali kolegija Baze podataka, Fakultet elektrotehnike i računarstva, ožujak 2021.
- [2] Nastavni materijali kolegija Razvoj primijenjene programske potpore, Fakultet elektrotehnike i računarstva, travanj 2021.
- [3] Nastavni materijali kolegija Programsko inženjerstvo, Fakultet elektrotehnike i računarstva, travanj 2021.
- [4] Baeldung, <https://www.baeldung.com>; pristupljeno 24. ožujka 2021.
- [5] Stack Overflow, <https://stackoverflow.com>; pristupljeno 2. travnja 2021.
- [6] Spring, <https://spring.io>; pristupljeno 20. Ožujka 2021.

Sažetak

U ovom završnom radu opisan je proces izrade programskog sustava za veleprodajnu tvrtku. Prvo je modelirana i implementirana baza podataka u PostgreSQL-u, a zatim i web aplikacija korištenjem programskih jezika Java i JavaScript, odnosno radnih okvira Spring Boot i React.js. Ostvareno je praćenje svih aspekata rada tvrtke, poput pohranjivanja, izmjene i brisanja podataka o zaposlenicima, proizvodima, poslovnim partnerima, računima, primkama te skladištima i stanjima skladišta. Osim navedenih funkcionalnosti, pružena je mogućnost analize poslovanja putem grafičkog prikaza statistika vezanih za promet grupiran po raznim kategorijama.

Ključne riječi: veleprodajna tvrtka, skladište, statistika, Java, Spring Boot, React.js, backend, frontend

Summary

This bachelor thesis describes the process of creating a software system for a wholesale company. The database was modeled and implemented in PostgreSQL, and then the web application using the Java and JavaScript programming languages, more precisely the Spring Boot and React.js frameworks. Monitoring of all aspects of the company's work was achieved, such as storing, modifying and deleting data on employees, products, business partners, invoices, receipts, and warehouses and warehouse balances. In addition to the above functionalities, the possibility of business analysis is provided through a graphical display of statistical data relating to turnover grouped by various categories.

Keywords: wholesale company, warehouse, statistics, Java, Spring Boot, React.js, backend, frontend