Baza podataka i web-aplikacija za praćenje osobnih financija

Grdić, Mateo

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1486

BAZA PODATAKA I WEB-APLIKACIJA ZA PRAĆENJE OSOBNIH FINANCIJA

Mateo Grdić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1486

BAZA PODATAKA I WEB-APLIKACIJA ZA PRAĆENJE OSOBNIH FINANCIJA

Mateo Grdić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1486

Pristupnik:	**Mateo Grdić (0036537685)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	prof. dr. sc. Boris Vrdoljak

Zadatak: **Baza podataka i web-aplikacija za praćenje osobnih financija**

Opis zadatka:

U sklopu ovog završnog rada potrebno je izraditi bazu podataka u kojoj će se pohranjivati podaci vezani za praćenje osobnih financija. 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 radnih okvira Django i React napraviti web-aplikaciju koja omogućava pregled, unos, brisanje i izmjenu podataka. Web-aplikacija treba prijavljenom korisniku omogućiti praćenje osobnih financija, što uključuje praćenje potrošnje i prihoda, raspodjelu novca po ciljevima ili potrebama te filtriranje i analizu podataka.

Rok za predaju rada: 14. lipnja 2024.

Sadržaj

Uvod

U današnjem dinamičkom ekonomskom okruženju, a i tehnološkom, upravljanje osobnim financijama postaje vrlo važna navika za pojedinca. S obzirom na brzinu promjena u financijskim tržištima, potreba za učinkovitim alatima za praćenje i upravljanje osobnim financijama nikada nije bila veća.

U tom kontekstu razvoj web aplikacije za praćenje osobnih financija predstavlja ključan korak u olakšavanju procesa upravljanja financijama pojedinaca. Aplikacija omogućava analizu troškova, prihoda i ulaganja s ciljem stvaranje svijesti i brige o financijama, te sam osobni napredak.

Aplikacija je izvedena kao web aplikacija koja se sastoji od tri ključna dijela: baze podataka, klijentske strane i serverske strane web aplikacije. Za bazu podataka, korišten je PostgreSQL, snažan i pouzdan sistem upravljanja bazama podataka. Za klijentsku stranu aplikacije korištena je React knjižnica, popularnu JavaScript knjižnica koja omogućuje brzo i efikasno kreiranje interaktivnih korisničkih sučelja. Za serversku stranu korišten je Pythonov radni okvir Django, koji pruža stabilnu i skalabilnu arhitekturu za razvoj i upravljanje serverske strane aplikacije. Kombinacija ovih tehnologija osigurava robusnu, sigurnu i intuitivnu platformu za praćenje osobnih financija.

U prvom dijelu rada detaljno su specificirani korisnički zahtjevi i slučajevi uporabe aplikacije. Drugi dio rada posvećen je opisu same baze podataka, dok se treći dio bavi implementacijskim pitanjima same aplikacije.

1. Specifikacija zahtjeva

1.1. Opis zadatka

Aplikacija za praćenje osobnih financija omogućuje korisnicima efikasno upravljanje svojim financijama putem web sučelja. Primarne funkcionalnosti uključuju evidentiranje i praćenje prihoda, troškova, investicija, postavljanje financijskih ciljeva, stvaranje zadataka kao i analizu financijskih aktivnosti. Korisnici se dijele na prijavljene, neprijavljene i administratore. Svaka od ovih uloga ima određene ovlasti i pristupne razine unutar aplikacije.

Analiza svih financijskih podataka omogućuje korisnicima bolje razumijevanje njihovih financijskih navika, identifikaciju trendova i postizanje financijskih ciljeva. Ova funkcionalnost pruža korisnicima vrijedne uvide i pomaže im u donošenju financijskih odluka.

1.2. Analiza zahtjeva

U analizi zahtjeva obrađeni su sladeći aspekti:

- Funkcionalni zahtjevi: Definirati detaljno sve funkcionalnosti koje aplikacija mora imati. Za analizu funkcionalnih zahtjeva korišten je dijagram slučajeva korištenja (engl. Use case diagram). Cjelovit opis slučajeva korištenja je opisan u poglavlju 1.3.
- Nefunkcionalni zahtjevi: upravljanje performansama, upravljanje podacima, sigurnosni i zakonski zahtjevi.

1.2.1. Funkcionalni zahtjevi

Aktori

- Neprijavljeni korisnik
- Prijavljeni korisnik
- Administrator

- Baza podataka (sustav)

U nastavku je naveden detaljan opis svih aktora.

Neprijavljeni korisnik

- Uloga: Posjetitelj aplikacije bez korisničkog računa ili prijavljenih vjerodajnica.
- Svrha: Registracija novog korisnika i prijava.
- Funkcionalni zahtjevi:
  - Pristup stranici za registraciju
  - Pristup stranici za prijavu

Prijavljeni korisnik

- Uloga: Registrirani korisnik aplikacije koji se prijavio sa svojim vjerodajnicama.
- Svrha: Upravljanje osobnim financijama putem web aplikacije.
- Funkcionalni zahtjevi:
  - Prijava i odjava
  - Pristup stranici za prijavu
  - Brisanje računa
  - Kreiranje, uređivanje, brisanje i pregled transakcija
  - Analiza transakcija putem filtriranja, pregleda statistika i grafova.
  - Kreiranje, uređivanje, uređivanje dodijeljenih sredstava i brisanje ciljeva.
  - Kreiranje, mijenjanje statusa i brisanje zadataka

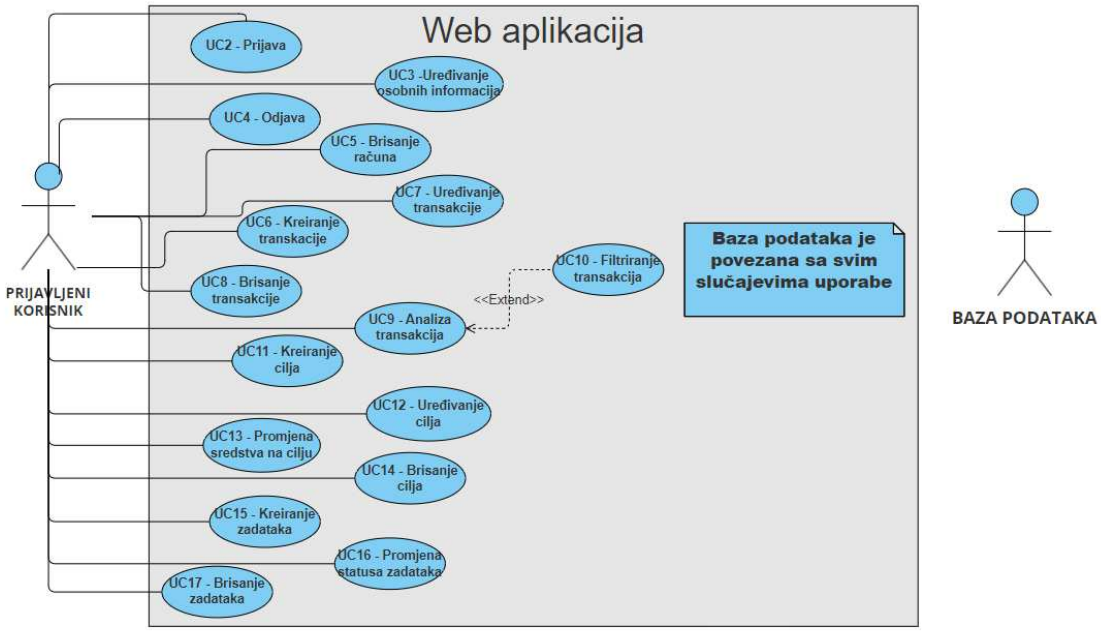
Administrator

- Uloga: Osoba s posebnim ovlastima za upravljanje svim podacima u sustavu.
- Svrha: Osiguravanje pravilnog funkcioniranja sustava i upravljanje podacima.
- Funkcionalni zahtjevi:
  - Pristup svim korisničkim podacima
  - Uređivanje i brisanje podataka putem Django REST radnog okvira

Baza podataka

- Uloga: Sustav za pohranu i upravljanje podacima aplikacije.
- Svrha: Osiguravanje pravilne pohrane, pristupa, ažuriranja i brisanje podataka.
- Funkcionalni zahtjevi:
  - Pohrana podataka o korisnicima, transakcijama, ciljevima i zadacima
  - Osiguravanje integriteta i sigurnosti podataka
  - Omogućavanje brzog i efikasnog pristupa podacima putem aplikacije
  - Izvršavanje upita za kreiranje, čitanje, ažuriranje i brisanje podataka

Na slici 1.1. prikazan je dijagram slučajeva korištenja, statički UML dijagram (engl. Unified Modeling Language), za prijavljenog korisnika.



Sl. 2.1 Dijagram slučajeva korištenja za prijavljenog korisnika

1.2.2. Nefunkcionalni zahtjevi

Upravljanje performansama – uključuje osiguravanje brzine odaziva sustava, minimalno vrijeme učitavanja i otpornost na visoko opterećenje. Sustav treba osigurati korisniku brz i učinkovit pristup svojim podacima. Ovo uključuje optimizaciju dohvaćanja stvari sa serverske strane, te implementaciju optimiziranih upita bazi podataka.

Upravljanje podacima – uključuje sigurnost skladištenje podatka korisno, redovito sigurnosno kopiranje, integritet podatak te dostupnost podataka kroz skalabilnu i pouzdanu bazu podataka. Važno je korisniku osigurati obradu podataka u skladu s pravilima privatnosti i zaštitom podataka.

Sigurnosni zahtjevi – uključuju implementaciju autentikacije i autorizacije korisnika, enkripciju podataka, sigurno upravljanje sesijama, zaštitu od poznatih sigurnosnih prijetnji poput SQL injekcija i XSS napada te redovite sigurnosne revizije i nadogradnje sustava.

Zakonski zahtjevi – uključuje da aplikacija za praćenje osobnih financija bude u skladu s relevantnim zakonima i propisima, uključujući pravila o zaštiti podataka kao što je Opća uredba o zaštiti podataka (GDPR). Ovo uključuje osiguranje transparentnosti u prikupljanju i obradi korisničkih podataka, poštivanje prava korisnika na privatnost, osiguranje pristanka korisnika za prikupljanje i obradu njihovih podataka te osiguranje mehanizama za brisanje ili ispravak osobnih podataka korisnika po zahtjevu.

1.3. Opis slučajeva korištenja

UC1 – Registracija

- Opis: Registracija omogućava korisnicima da stvore novi račun u aplikaciji.
- Sudionici: Neprijavljeni korisnik, sustav
- Preduvjet: Korisnik ne smije biti prijavljen i mora pristupiti stranici za registraciju.
- Postuvjet: Korisnik uspješno registriran.
- Osnovni scenarij događanja:
    1. Korisnik pristupa stranici za registraciju.
    2. Korisnik unosi svoje korisničko ime, e-mail adresu, lozinku i početni saldo.
    3. Korisnik potvrđuje registraciju klikom na odgovarajući gumb.
    4. Sustav provjerava unesene podatke i kreira novi korisnički račun.
    5. Korisnik je preusmjeren na stranicu za prijavu.

UC2 – Prijava

- Opis: Prijava omogućava registriranim korisnicima pristup aplikaciji.
- Sudionici: Neprijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora imati registriran račun i ne smije biti prijavljen.
- Postuvjet: Korisnik uspješno prijavljen na aplikaciju.

- Osnovni scenarij događanja:
    1. Korisnik pristupa stranici za prijavu.
    2. Korisnik unosi svoje korisničko ime i lozinku.
    3. Korisnik potvrđuje prijavu klikom na odgovarajući gumb.
    4. Sustav provjerava unesene podatke i prijavljuje korisnika na aplikaciju.
    5. Korisnik je preusmjeren na početnu stranicu.

UC3 – Uređivanje osobnih informacija

- Opis: Omogućava korisnicima promjenu korisničkih podataka.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Podaci korisnika su uspješno ažurirani.
- Osnovni scenarij događanja:
    1. Korisnik pristupa svojem korisničkom profilu.
    2. Korisnik mijenja željene podatke kao što su ime, prezime, adresa, kontakt informacije itd.
    3. Korisnik potvrđuje promjene klikom na odgovarajući gumb.
    4. Sustav provjerava ispravnost unesenih podataka i ažurira korisnički profil.

UC4 – Odjava

- Opis: Omogućava korisnicima da se odjave iz aplikacije.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Korisnik je uspješno odjavljen i sesija je završena.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za odjavu.
    2. Sustav završava trenutnu sesiju korisnika i preusmjerava ga na početnu stranicu.

UC5 – Brisanje računa

- Opis: Omogućava korisnicima trajno uklanjanje svog korisničkog računa iz sustava.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Korisnikov račun je uspješno obrisan iz sustava.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za brisanje računa.
    2. Korisnik potvrđuje brisanje računa.
    3. Sustav provjerava valjanost korisnikovih podataka i potvrđuje brisanje računa.

4. Svi podaci povezani s korisnikovim računom se trajno brišu iz sustava.

UC6 – Kreiranje transakcije

- Opis: Omogućava korisnicima stvaranje novih financijskih transakcija.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Nova transakcija je uspješno dodana u sustav.
- Osnovni scenarij događanja:
    1. Korisnik pristupa stranici vrste transakcije koju želi stvoriti.
    2. Korisnik pristupa opciji za kreiranje nove transakcije.
    3. Korisnik unosi detalje transakcije, kao što su iznos, opis itd.
    4. Korisnik potvrđuje kreiranje transakcije klikom na odgovarajući gumb.
    5. Sustav provjerava ispravnost unesenih podataka i dodaje novu transakciju u bazu podataka.

UC7 – Uređivanje transakcije

- Opis: Omogućava korisnicima izmjenu postojećih financijskih transakcija.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeću transakciju.
- Postuvjet: Izmjene na transakciji su uspješno spremljene.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za uređivanje transakcije.
    2. Korisnik odabire transakciju koju želi urediti.
    3. Korisnik mijenja detalje transakcije, kao što su iznos, datum, opis itd.
    4. Korisnik potvrđuje izmjene klikom na odgovarajući gumb.
    5. Sustav provjerava ispravnost unesenih podataka i ažurira transakciju u bazi podataka.

UC8 – Brisanje transakcije

- Opis: Omogućava korisnicima trajno uklanjanje postojećih financijskih transakcija.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeću transakciju.
- Postuvjet: Transakcija je uspješno obrisana iz sustava.
- Osnovni scenarij događanja:
    1. Korisnik odabire transakciju koju želi izbrisati.
    2. Korisnik potvrđuje brisanje transakcije klikom na odgovarajući gumb.
    3. Sustav trajno uklanja transakciju iz baze podataka.

UC9 – Analiza transakcija

- Opis: Analiza transakcija omogućava korisnicima pregled i analizu svojih financijskih transakcija.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeću transakciju.
- Postuvjet: Korisnik je uspješno pregledao i analizirao svoje transakcije putem tablice i grafova.
- Osnovni scenarij događanja:
    1. Korisnik pristupa određenoj vrsti transakcija.
    2. Korisnik pregledava podatke iz tablice i grafova.

UC10 – Filtriranje transakcija

- Opis: Omogućava korisnicima filtriranje svojih financijskih transakcija prema određenim kriterijima.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeće transakcije.
- Postuvjet: Korisnik je uspješno filtrirao i pregledao transakcije prema svojim kriterijima.
- Osnovni scenarij događanja:
    1. Korisnik pristupa određenoj vrsti transakcija.
    2. Korisnik odabire željene kriterije filtriranja, kao što su datum, iznos, kategorija transakcije itd.
    3. Korisnik primjenjuje filtre na svoje transakcije klikom na odgovarajući gumb.
    4. Sustav prikazuje korisniku filtrirane transakcije prema odabranim kriterijima.

UC11 – Kreiranje cilja

- Opis: Kreiranje cilja omogućava korisnicima postavljanje financijskih ciljeva koje žele postići.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Novi cilj je uspješno dodan u sustav.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za postavljanje novog cilja.
    2. Korisnik unosi detalje cilja, kao što su naziv cilja, iznos koji želi uštedjeti, rok za postizanje cilja itd.

3. Korisnik potvrđuje kreiranje cilja klikom na odgovarajući gumb.
4. Sustav provjerava ispravnost unesenih podataka i dodaje novi cilj u bazu podataka.

UC12 – Uređivanje cilja

- Opis: Uređivanje cilja omogućava korisnicima izmjenu postojećih financijskih ciljeva.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeći cilj.
- Postuvjet: Promjene na cilju su uspješno spremljene.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za uređivanje cilja.
    2. Korisnik odabire cilj koji želi urediti.
    3. Korisnik mijenja detalje cilja, kao što su naziv cilja, iznos koji želi uštedjeti, rok za postizanje cilja.
    4. Korisnik potvrđuje izmjene klikom na odgovarajući gumb.
    5. Sustav provjerava ispravnost unesenih podataka i ažurira cilj u bazi podataka.

UC13 – Promjena sredstava na cilj

- Opis: Omogućava korisnicima promjenu iznosa uplaćenog novca na svoje financijske ciljeve.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeći cilj.
- Postuvjet: Novi iznos je uspješno postavljen na cilj
- Osnovni scenarij događanja:
    1. Korisnik odabire cilj na kojem želi promijeniti svotu prikupljenog novca.
    2. Korisnik unosi iznos za koji želi promijeniti.
    3. Korisnik potvrđuje uplatu klikom na odgovarajući gumb.
    4. Sustav provjerava ispravnost unesenih podataka i ažurira stanje cilja u bazi podataka.

UC14 – Brisanje cilja

- Opis: Omogućava korisnicima uklanjanje postojećih financijskih ciljeva.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeći cilj.
- Postuvjet: Cilj je uspješno obrisan iz sustava.
- Osnovni scenarij događanja:
    1. Korisnik odabire cilj koji želi izbrisati.
    2. Korisnik potvrđuje brisanje cilja klikom na odgovarajući gumb.
    3. Sustav trajno uklanja cilj iz baze podataka.

UC15 – Kreiranje zadataka

- Opis: Omogućava korisnicima stvaranje novih obaveza koje žele obaviti.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju.
- Postuvjet: Novi zadatak je uspješno dodan u sustav.
- Osnovni scenarij događanja:
    1. Korisnik pristupa opciji za kreiranje novog zadatka.
    2. Korisnik unosi detalje zadatka, kao što su naziv zadatka, opis, rok za izvršenje itd.
    3. Korisnik potvrđuje kreiranje zadatka klikom na odgovarajući gumb.
    4. Sustav provjerava ispravnost unesenih podataka i dodaje novi zadatak u bazu podataka

UC16 – Promjena statusa zadataka

- Opis: Omogućava korisnicima promjenu statusa zadatku.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeći zadatak.
- Postuvjet: Zadatku je promijenjen status.
- Osnovni scenarij događanja:
    1. Korisnik pristupa svojem popisu zadataka.
    2. Korisnik odabire zadatak koji želi označiti kao obavljen.
    3. Korisnik označava zadatak kao obavljen klikom na odgovarajući gumb.
    4. Sustav ažurira status zadatka u bazi podataka.

UC17 – Brisanje zadataka

- Opis: Omogućava korisnicima uklanjanje postojećih zadataka.
- Sudionici: Prijavljeni korisnici, sustav.
- Preduvjet: Korisnik mora biti prijavljen na aplikaciju i imati postojeći zadatak.
- Postuvjet: Zadatak je uspješno obrisan iz sustava.
- Osnovni scenarij događanja:
    1. Korisnik odabire zadatak koji želi izbrisati.
    2. Korisnik potvrđuje brisanje zadatka klikom na odgovarajući gumb.
    3. Sustav trajno uklanja zadatak iz baze podataka.

UC18 – Uređivanje svih podataka

- Opis: Administrator može uređivati sve podatke u sustavu putem sučelja Django REST radnog okvira.
- Sudionici: Administrator, sustav.
- Preduvjet: Administrator mora biti prijavljen na aplikaciju s administratorskim ovlastima.

- Postuvjet: Podaci u sustavu su uspješno izmijenjeni.
- Osnovni scenarij događanja:
    1. Administrator pristupa sučelju Django REST radnog okvira.
    2. Administrator odabire podatke koje želi urediti (npr. korisničke podatke, transakcije, ciljeve, zadatke).
    3. Administrator mijenja potrebne informacije.
    4. Administrator potvrđuje izmjene putem sučelja.
    5. Sustav provjerava ispravnost izmjena i ažurira podatke u bazi podataka.

UC19 – Brisanje svih podataka

- Opis: Administrator može brisati sve podatke u sustavu putem sučelja Django REST radnog okvira.
- Sudionici: Administrator, sustav.
- Preduvjet: Administrator mora biti prijavljen na aplikaciju s administratorskim ovlastima.
- Postuvjet: Podaci u sustavu su uspješno obrisani.
- Osnovni scenarij događanja:
    1. Administrator pristupa sučelju Django REST radnog okvira.
    2. Administrator odabire podatke koje želi obrisati (npr. korisničke podatke, transakcije, ciljeve, zadatke).
    3. Administrator potvrđuje brisanje podataka putem sučelja.
    4. Sustav trajno uklanja odabrane podatke iz baze podataka.

2. Baza podataka

2.1. Korištene tehnologije

Za implementaciju baze podataka korištena je tehnologija PostgreSQL. PostgreSQL je napredan, open-source sustav za upravljanje relacijskim bazama podataka poznat po svojoj pouzdanosti, robusnosti i bogatstvu funkcionalnosti. Ključne značajke PostgreSQL-a uključuju: ACID usklađenost, podrška za složene upite i transakcije, proširivost, podrška za JSON i hibridne modele te sigurnost.

Važno je napomenuti da su sve tablice u bazi podataka kreirane putem serverske strane koja je implementirana u Djangu, što je popularni Python web radni okvir o kojem je više rečeno u poglavlju 3.1. Korištene tehnologije. Korištenjem Django ORM-a (Object-Relational Mapping), definirane su strukture tablica i relacija između njih, čime je osigurana integracija između Django aplikacije i PostgreSQL baze podataka. Django ORM također pruža moćan alat za pristup podacima, omogućava jednostavan i intuitivan način postavljanja upita i manipulacije podacima unutar baze podataka.

2.2. ER model

ER model prikazuje osnovne entitete u bazi podataka i njihove međusobne odnose. U ovom slučaju, ER model uključuje entitete kao što su korisnici, financijske transakcije, investicije, financijski ciljevi i zadaci. Cjelokupni ER dijagram prikazan je na slici 2.1.

Slijedi popis entiteta i njihovih atributa iz ER modela:

- **Korisnik** = korisnikId, korisnickoIme, lozinka, email, pocetniSaldo, saldo, datumPridruzivanja

  Entitet **Korisnik** predstavlja korisnika sustava. Svaki korisnik ima jedinstveni identifikator, jedinstveno korisničko ime, lozinku, jedinstvenu email adresu, početni saldo, trenutni saldo te datum pridruživanja.

- **Transakcija =** transakcijaId, transakcijaOpis, transakcijaIznos, transakcijaDatum, transakcijaVrsta

Entitet **Transakcija** predstavlja transakciju koju korisnik obavlja. Svaka transakcija ima jedinstveni identifikator, opcionalni opis, iznos transakcije u eurima, vrstu transakcije te datum kad je transakcija izvršena.

- **Trosak=** transakcijaId, trosakKategorija

  Entitet **Trosak** predstavlja troškove. Svaki trošak ima jedinstveni identifikator koji je strani ključ koji referencira istoimeni atribut u relaciji **Transakcija** te enum koji predstavlja kategoriju troška.

- **Prihod** = transakcijaId, prihodiKategorija

  Entitet **Prihod** predstavlja prihode. Svaki prihod ima jedinstveni identifikator koji je strani ključ koji referencira istoimeni atribut u relaciji **Transakcija** te enum koji predstavlja kategoriju prihoda.

- **Investicija** = transakcijaId, investicijaNaslov, investicijaKategorija, godisnjiPovrat

  Entitet **Investicija** predstavlja investicije. Svaka investicija ima jedinstveni identifikator koji je strani ključ koji referencira istoimeni atribut u relaciji **Transakcija**, naslov, enum koji predstavlja kategoriju investicije, te opcionalni atribut koji predstavlja prosječni očekivani godišnji povrat investicije.

- **Zadatak** = zadatakId, prioritet, zadatakOpis, zadatakNaslov, zadatakStatus, zadatakDatumRoka

  Entitet **Zadatak** predstavlja zadatke koje korisnik ima. Svaki zadatak ima jedinstveni identifikator, prioritet, opcionalni opis, naslov, status, te datum rok izvršenja.

- **Cilj** = ciljId, ciljNaslov, ciljniIznos, trenutniIznos, ciljDatumRoka

  Entitet **Cilj** predstavlja ciljeve koje korisnik postavlja. Svaki cilj ima jedinstveni identifikator, naslov, ciljani iznos, trenutni iznos, te datum roka.

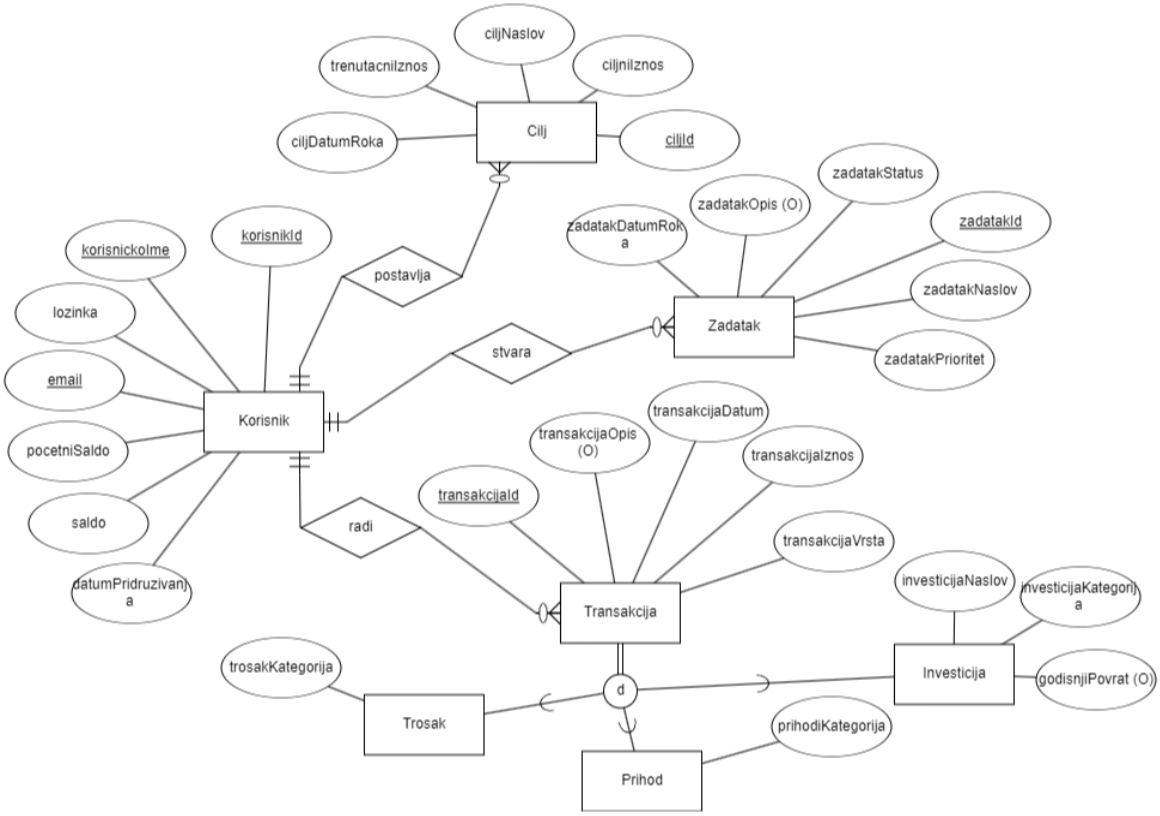
Slijedi popis veza iz ER modela:

- **radi** = transakcijaId, korisnikId

  Veza **radi** povezuje transakcije s korisnicima. Svaka transakcija pripada određenom korisniku, korisnik može imati više transakcija.

- **postavlja** = <u>ciljId</u>, korisnikId
- Veza **postavlja** povezuje ciljeve s korisnicima. Svaki korisnik može imati postavljen više ciljeva, a svaki cilj pripada jednom korisniku.
- **stvara** = <u>zadatakId</u>, korisnikId

  Veza **stvara** povezuje zadatke s korisnicima. Svaki zadatak pripada određenom korisniku.



Sl. 2.1 ER model baze podataka

2.3. Relacijski model

Relacijski model prikazuje tablice u bazi podataka s njihovim stupcima i relacijama između tih tablica. Ovaj relacijski model je preslika gore navedenog ER modela. Cjelokupni ER dijagram prikazan je na slici 2.2.

Slijedi popis relacija:

Zeleno obojena ćelija tablice označava primarni ključ.

Plavo obojena ćelija tablice označava strani ključ.

Korisnik

Relacija Korisnik sadrži atribute korisnikId, email, korisnickoIme, saldo, pocetniSaldo, lozinka i datumPridruzivanja. Detaljan opis atributa prikazan je u tablici 2.3.1.

Tablica 2.3.1 Relacija Korisnik

Atribut	**Tip podatka**	**Opis**
korisnikId	INT	Primarni ključ tablice, dodjeljuje se automatski
email	VARCHAR	Jedinstveni email korisnika
korisnickoIme	VARCHAR	Jedinstveno ime koje korisnik odabere tijekom registracije
saldo	FLOAT	Trenutno stanje računa korisnika
pocetniSaldo	FLOAT	Početno stanje računa korisnika
lozinka	VARCHAR	Lozinka korisnika (hash)
datumPridruzivanja	DATE	Datum kada se korisnik pridružio

Transakcija

Relacija Transakcija sadrži atribute transakcijaId, korisnikId, transakcijaIznos, transakcijaDatum, transakcijaOpis, transakcijaVrsta. Detaljan opis atributa prikazan je u tablici 2.3.2.

Tablica 2.3.2 Relacija Transakcija

Atribut	**Tip podatka**	**Opis**
transakcijaId	INT	Primarni ključ tablice, dodjeljuje se automatski
korisnikId	INT	Strani ključ iz tablice **Korisnik** koji se odnosi na njegov ID
transakcijaIznos	FLOAT	Iznos transakcije

transakcijaDatum	DATE	Datum kad je transakcija obavljena
transakcijaOpis	VARCHAR	Kratki opis transakcije
transakcijaVrsta	VARCHAR	Enum s vrijednostima koje pristavljaju investiciju, trošak i prihod

Investicija

Relacija Investicija sadrži atribute transakcijaId, investicijaKategorija, godisnjiPovrat, investicijaNaslov. Detaljan opis atributa prikazan je u tablici 2.3.3.

Tablica 2.3.3 Relacija Investicija

Atribut	**Tip podatka**	**Opis**
transakcijaId	INT	Strani ključ iz tablice **Transakcija** koji se odnosi na njezin ID
investicijaKategorija	VARCHAR	Enum s deset definiranih vrijednosti
godisnjiPovrat	FLOAT	Opcionalni, očekivani godišnji povrat
investicijaNaslov	VARCHAR	Naslov investicije

Trošak

Relacija Trošak sadrži atribute transakcijaId i trosakKategorija. Detaljan opis atributa prikazan je u tablici 2.3.4.

Tablica 2.3.4 Relacija Trošak

Atribut	**Tip podatka**	**Opis**
transakcijaId	INT	Strani ključ iz tablice **Transakcija** koji se odnosi na njezin ID
trosakKategorija	VARCHAR	Enum s deset definiranih vrijednosti

Prihod

Relacija Prihod sadrži atribute prihod i prihodKategorija. Detaljan opis atributa prikazan je u tablici 2.3.5.

Tablica 2.3.5 Relacija Prihod

Atribut	**Tip podatka**	**Opis**
transakcijaId	INT	Strani ključ iz tablice **Transakcija** koji se odnosi na njezin ID
prihodKategorija	VARCHAR	Enum s deset definiranih vrijednosti

Cilj

Relacija Cilj sadrži atribute ciljId, korisnikId, trenutniIznos, ciljniIznos, ciljNaslov ciljDatumRok. Detaljan opis atributa prikazan je u tablici 2.3.6.

Tablica 2.3.6 Relacija Cilj

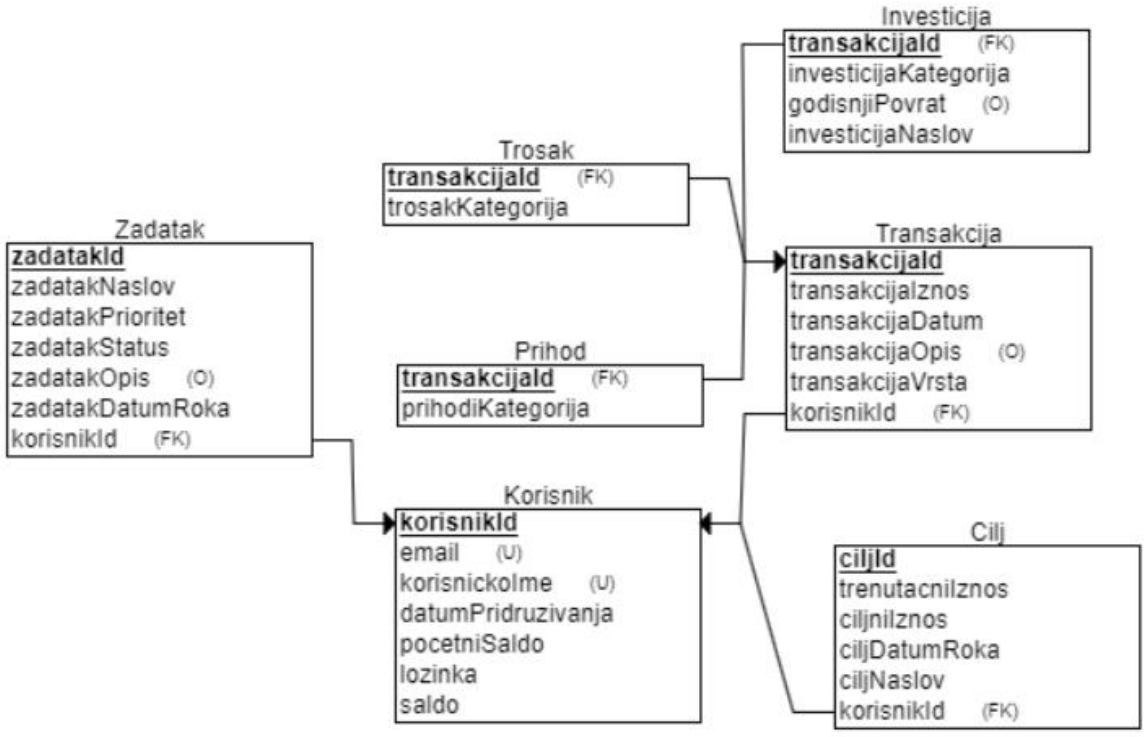
Atribut	**Tip podatka**	**Opis**
ciljId	INT	Primarni ključ tablice, dodjeljuje se automatski
korisnikId	INT	Strani ključ iz tablice **Korisnik** koji se odnosi na njegov ID
trenutniIznos	FLOAT	Trenutni prikupljeni iznos
ciljniIznos	FLOAT	Ciljni iznos koji treba prikupiti
ciljDatumRok	DATE	Datum do kada cilj treba biti postignut
ciljNaslov	VARCHAR	Naslov cilja

Zadatak

Relacija Zadatak sadrži atribute zadatakId, korisnikId, zadatakNaslov, zadatakPrioritet, zadatakStatus, zadatakOpis, zadatakDatumRok. Detaljan opis atributa prikazan je u tablici 2.3.7.

Tablica 2.3.7 Relacija Zadatak

Atribut	**Tip podatka**	**Opis**
zadatakId	INT	Primarni ključ tablice, dodjeljuje se automatski
korisnikId	INT	Strani ključ iz tablice **Korisnik** koji se odnosi na njegov ID
zadatakNaslov	VARCHAR	Naslov zadatka
zadatakPrioritet	VARCHAR	Enum s vrijednostima: A, B, C, D
zadatakStatus	BOOLEAN	Status zadatka (završen ili ne završen)
zadatakOpis	VARCHAR	Opcionalan opis zadatka
zadatakDatumRok	DATE	Datum do kada zadatak treba biti završen



Sl. 2.2 Relacijski model

3. Implementacija web aplikacije

3.1. Korištene tehnologije

Web aplikacija je implementirana koristeći nekoliko ključnih tehnologija, koje zajedno omogućuju razvoj robusne, interaktivne i skalabilne aplikacije. U nastavku je pregled svake od korištenih tehnologija. Kombinacija HTML-a, CSS-a, JavaScripta i Pythona omogućuje stvaranje robusne i interaktivne web aplikacije. HTML pruža osnovnu strukturu, CSS definira izgled i stil, JavaScript dodaje interaktivnost, dok Python (Django) upravlja serverskom logikom i bazom podataka. Korištenje ovih tehnologija zajedno osigurava skalabilnu, održivu i korisnički pristupačnu aplikaciju.

3.1.1. HTML

HTML (HyperText Markup Language) je standardni označni jezik za izradu web stranica i web aplikacija. Njegova glavna funkcija je strukturiranje sadržaja na webu. HTML koristi oznake (tagove) za definiranje različitih elemenata na stranici kao što su naslovi, paragrafi, liste, linkovi, slike i drugi multimedijalni sadržaji. Ključne značajke su da pruža strukturiran sadržaj, multimedijalnu podršku i dobru integraciju s drugim tehnologijama kao što su CSS i JavaScript.

3.1.2. CSS

CSS (Cascading Style Sheets) je jezik za opisivanje stila HTML dokumenta. Koristi se za definiranje izgleda i formatiranja web stranica, omogućujući razdvajanje sadržaja od prezentacije. Ključne značajke su da omogućava stilizaciju, responzivni dizajn, modularnost te animacije.

3.1.3. JavaScript

JavaScript je dinamički programski jezik koji se koristi za dodavanje interaktivnosti i dinamičkih funkcionalnosti web stranicama. Izvodi se na klijentskoj strani, što omogućuje brzu i responzivnu interakciju s korisnicima. Omogućuje interaktivnost, manipulaciju DOM-om (promjena strukture, sadržaja i stila web stranice u stvarnom vremenu).

JavaScript ima vrlo široku uporabu, ne koristi se samo za razvoj klijentske strane nego se može koristiti za backend razvoj (Node.js), mobilne aplikacije (React Native), desktop aplikacije (Electron) i još mnogo toga. U kontekstu ove aplikacije korišten je React, popularna JavaScript knjižnica za izgradnju korisničkih sučelja. React omogućuje izradu kompleksnih i brzih korisničkih sučelja pomoću komponenata koje se lako mogu ponovno koristiti.

3.1.4. Python

Python je programski jezik visoke razine, interpretirani programski jezik poznat po svojoj čitljivosti i jednostavnosti. U kontekstu ove web aplikacije, Python se koristi za razvoj serverske strane koristeći Django radni okvir. Značajke Pythona su da je vrlo jednostavan, čitljiv, raznovrstan u svojoj primjeni te bogat je već gotovim bibliotekama i okvirima. Koristi se u raznim područjima uključujući razvoj web aplikacija, strojnome učenju, testiranju aplikacija, obradom podataka i još mnogim područjima.

3.2. Arhitektura aplikacije

Aplikacija je izvedena kao web aplikacija koja se sastoji od tri ključna dijela:

Baza podataka: PostgreSQL baza podataka pohranjuje sve podatke aplikacije, uključujući informacije o korisnicima, financijskim transakcijama, investicijama i ciljevima.

Klijentska strana: React aplikacija pruža korisničko sučelje kroz koje korisnici mogu unositi i pregledavati svoje financijske podatke. React omogućava brzo i interaktivno korisničko iskustvo zahvaljujući svojoj komponentnoj arhitekturi.

Serverska strana: Django poslužitelj upravlja poslovnom logikom aplikacije i posredovanjem između klijentske strane i baze podataka. Django API osigurava da se svi zahtjevi korisnika pravilno obrađuju i da se podaci sigurno prenose.

Kombinacija arhitekture s tri sloja omogućuje jasnu podjelu odgovornosti, olakšava održavanje i proširivanje aplikacije te doprinosi skalabilnosti i robusnosti sistema. Svaki sloj je neovisan entitet, što omogućuje paralelan rad, lakše testiranje i izolaciju grešaka. Ova modularnost također omogućuje brže prilagodbe i reakcije na promjene, dok neovisno skaliranje svakog sloja omogućuje prilagodbu rastućim zahtjevima i povećava otpornost aplikacije na kvarove.

3.2.1. Prezentacijski sloj

Prezentacijski sloj je sloj aplikacije koji je direktno izložen korisnicima i omogućuje im interakciju s aplikacijom. U ovom sloju se nalazi korisničko sučelje (UI), koje može biti web sučelje, mobilna aplikacija ili neki drugi oblik sučelja prema korisniku. Prezentacijski sloj komunicira s ostalim slojevima aplikacije kako bi dohvaćao, prikazivao i ažurirao podatke prema korisničkim zahtjevima. Osim toga, prezentacijski sloj također može biti odgovoran za validaciju korisničkih unosa i upravljanje korisničkim sesijama. U kontekstu ove aplikacije, za implementaciju prezentacijskog sloja korišten je React, popularna JavaScript knjižnica za izgradnju korisničkih sučelja. Primjeri tehnologija koje se često koriste u prezentacijskom sloju su HTML, CSS, JavaScript, React, Angular, Vue.js i slično.

3.2.2. Logički sloj

Logički sloj je središnji dio aplikacije koji obavlja poslovnu logiku i obradu podataka. Ovaj sloj prima zahtjeve od prezentacijskog sloja, obrađuje ih, te komunicira s bazom podataka radi dohvaćanja i ažuriranja podataka. Uloga logičkog sloja je da provjeri ispravnost i konzistentnost podataka, primjeni poslovna pravila i pruži odgovore na zahtjeve prezentacijskom sloju. Ovdje se obično nalazi veći dio poslovne logike, kao što su algoritmi, procesi autentikacije i autorizacije, validacija podataka i sl. U kontekstu ove aplikacije, logički sloj implementiran je korištenjem Django Rest radnog okvira u kombinaciji s Pythonom. Ova kombinacija omogućuje obradu zahtjeva, validaciju podataka, autentikaciju i autorizaciju korisnika, te komunikaciju s bazom podataka radi dohvaćanja i ažuriranja podataka. Dodatno, logički sloj koristi Django serijalizator za pretvorbu objekata Pythona u JSON format pri slanju podataka prema prezentacijskom sloju, kao i za obradu podataka dobivenih od prezentacijskog sloja.

3.2.3. Sloj za pohranu podataka

Sloj za pohranu podataka je komponenta aplikacije koja je odgovorna za trajno čuvanje podataka. Ovaj sloj obično koristi bazu podataka ili neki drugi oblik trajnog skladišta podataka kako bi omogućio dugoročno čuvanje i organizaciju podataka. U kontekstu ove aplikacije, sloj za pohranu podataka koristi PostgreSQL, koji je relacijska baza podataka. PostgreSQL omogućuje pohranu strukturiranih podataka u obliku tablica i omogućuje

kompleksne upite i transakcije nad tim podacima. Ovaj sloj je odvojen od ostalih slojeva aplikacije i pruža neovisno skladište podataka koje logički sloj koristi za dohvaćanje, ažuriranje i brisanje podataka prema potrebi.

3.2.4. Komunikacija između slojeva

U ovom slučaju, komunikacija između slojeva odvija se na sljedeći način:

Komunikacija između prezentacijskog i logičkog sloja:

Prezentacijski sloj, koji koristi React, šalje zahtjeve logičkom sloju za obradu. Zahtjevi se prenose preko HTTP protokola, koristeći Fetch API.

Logički sloj, implementiran pomoću Django Rest radnog okvira, prima te zahtjeve i obrađuje ih prema poslovnoj logici aplikacije.

Odgovor logičkog sloja je obliku JSON-a.

Komunikacija između logičkog sloja i sloja za pohranu podataka:

Logički sloj, putem ORM-a (Object-Relational Mapping), komunicira s PostgreSQL bazom podataka.

ORM omogućuje logičkom sloju da izrazi zahtjeve za dohvaćanje, ažuriranje, dodavanje ili brisanje podataka koristeći Python objekte.

ORM prevodi ove zahtjeve u SQL upite koji se izvršavaju na PostgreSQL bazi podataka.

Rezultati ovih upita se zatim vraćaju logičkom sloju, koji ih dalje obrađuje i prosljeđuje prezentacijskom sloju.

Upravljanje korisničkim sesijama i autentikacijom:

Za upravljanje korisničkim sesijama i autentikacijom, logički sloj koristi sustav autentikacije Django Rest radnog okvira, koji može koristiti različite metode autentikacije poput tokena, sesija ili JWT-a (JSON Web Tokens).

U ovom slučaju za upravljanje korisničkim sesijama i autentikacijom uključuje korištenje JWT za autentikaciju. JWT token se automatski obnavlja svakih 1 sat (60 minuta). Nakon uspješne autentikacije, korisnički JWT token šalje se sljedećim zahtjevima prezentacijskog sloja, omogućujući pristup zaštićenim resursima na temelju korisničkih ovlasti. Ovaj mehanizam automatskog obnavljanja JWT tokena svakih 1 sat osigurava sigurnost i pouzdanost autentikacije korisnika u aplikaciji.

Kroz ovu strukturu komunikacije, svaki sloj aplikacije ima jasno definirane uloge i odgovornosti, što doprinosi modularnosti, skalabilnosti i održivosti cjelokupne aplikacije.

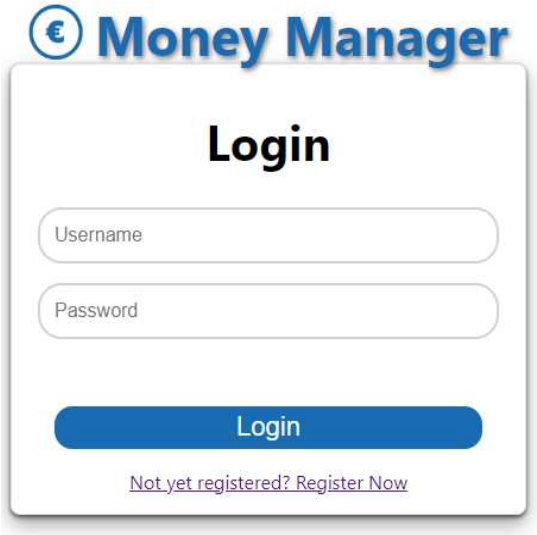
3.3. Korisničke upute

3.3.1. Korisnik

Korisničko sučelje omogućava korisniku prijavu, registraciju i pristup raznim funkcionalnostima aplikacije. Nakon uspješne prijave, korisnici mogu upravljati svojim financijama kroz pregled, stvaranje novih, uređivanje i brisanje podataka o transakcijama, ciljevima i zadacima. Korisnici također mogu upravljati svojim osobnim informacijama putem profila.

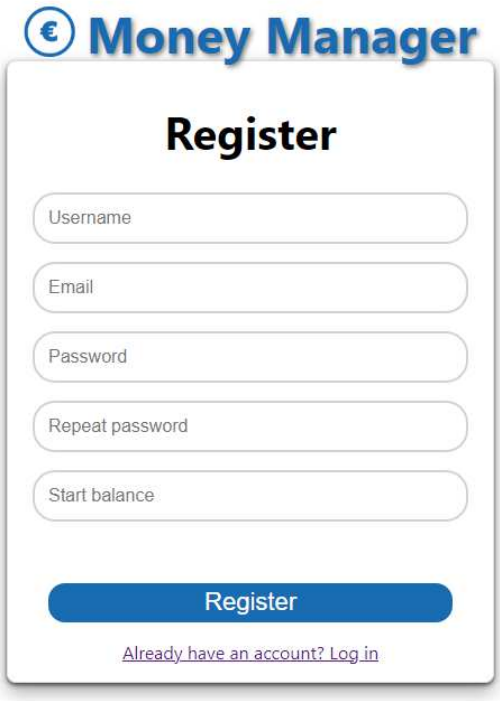
Prikaz forme za prijavu i registraciju

Korisnici pristupaju aplikaciji putem stranice za prijavu i registraciju. Na ovoj stranici korisnici mogu unijeti svoje korisničko ime i lozinku za prijavu, prikazano slikom 3.3.1, ili kliknuti na opciju za registraciju novog računa, prikazano slikom 3.3.2.



Sl. 3.3.1 Forma za prijavu

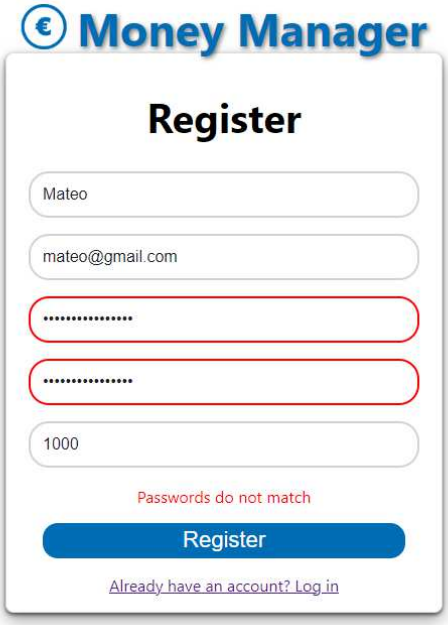
Korisnik unosi svoje korisničko ime i lozinku, zatim klikom na gumb „Login“ pokrećete akciju prijave. Klikom na „Register Now“ aplikacija otvara formu za registraciju novih korisnika.



Sl. 3.3.2 Forma za registraciju

Unosom korisničkog imena, emaila, lozinke, početnog salda (opcionalno) te klikom na gumb „Register“ stvara se novi korisnički račun. Klikom na „Log in“ otvara se forma za prijavu.

Ako dođe do greške, ispisuje se poruka o grešci, a polje koje je izazvalo grešku postaje crveno. Primjer prikazan slikom 3.3.3.



Sl. 3.3.3 Primjer pogreške pri registraciji

Početni zaslon

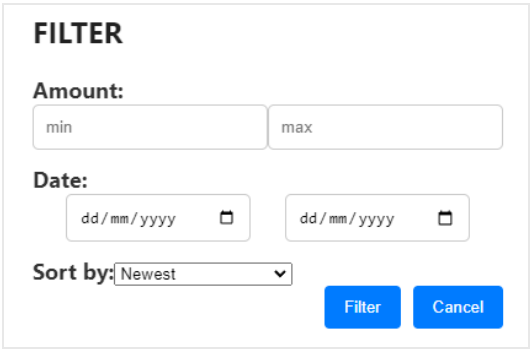
Nakon uspješne prijave i potvrde digitalnog identiteta, prikazuje se početna stranica „Dashboard“ prikazana slikom 3.3.4. Navigacijski bar nalazi se lijevo, prikazuje trenutnu poziciju na web aplikaciji i ostale mogućnosti, kao što su „General“, „Expenses“, „Income“, „Investments“, „Goals“, „ToDo“. U gornjem desnom kutu nalazi se gumb s korisničkim imenom koji vodi na prikaz informacija o profilu te gumb za odjavu.

Sl. 3.3.4 Prikaz početne stranice

„BALANCE“ prikazuje trenutno stanje salda. Tortni dijagram nudi mogućnost grupiranog prikaza po kategorijama troškova ili prihoda uz određeni vremenski filter. Također prikazana su 3 nadolazeća zadataka, te 3 cilja s najvećim postotkom prikupljenog iznosa.

Pregled transakcija

Odabirom opcije „General“ u navigacijskom baru prikazuje se zajednički prikaz svih transakcija, prikazan slikom 3.3.6. Transakcije se mogu filtrirati i sortirati po iznosu i vremenu stvaranja, slika 3.3.5 prikazuje filter.



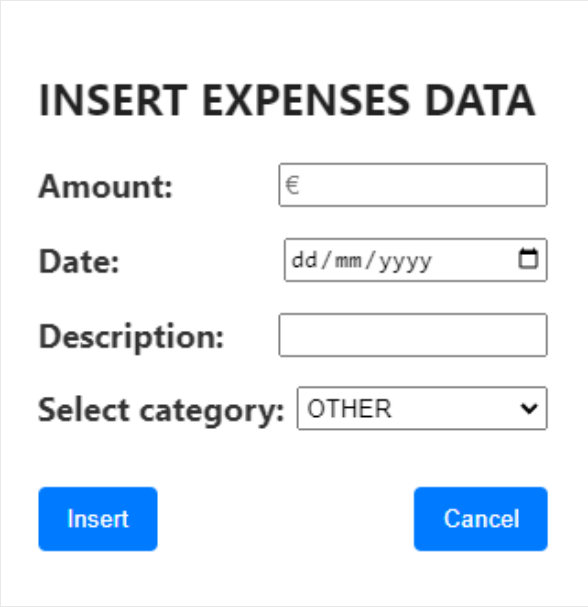
Sl. 3.3.5 Filter transakcija

Sl. 3.3.6 Prikaz pregleda transakcija

Kroz stupčasti graf prikazani su odnosi između pojedinih vrsta transakcija. Linijski graf nudi mogućnost vremenskog grupiranja po danu, mjesecu i godini te njihov prikaz za svaku vrstu. Tablica omogućuje prikaz detaljnih podataka o transakcijama, brisanje pojedinih transakcija te paginaciju za lakšu navigaciju kroz veće količine podataka.

Rashodi

Odabirom opcije „Expenses“ u navigacijskom baru prikazuju se svi troškovi i informacije o njima, prikazano slikom 3.3.8. Uz sve mogućnosti s prethodne stranice nudi dodatne mogućnost kao što su kreiranja i uređivanja troškova, prikazano na slici 3.3.7.



Sl. 3.3.7 Forma za unos novih troškova

Za izradu novog troška potrebno je unijeti iznos, datum i kategoriju, dok je opis opcionalan. Kategorije koje su ponuđene su za izradu troškova:

- Hobiji
- Zabava
- Odjeća
- Osiguranje
- Obrazovanje
- Zdravlje
- Režije
- Prijevoz
- Stanovanje
- Hrana
- Ostalo

Sl. 3.3.8 Prikaz troškova

Nakon stvaranja, brisanja ili uređivanja troška poruka s informacijom o akciji se pokaže ispod stupčastog grafa.

Prihodi

Odabirom opcije „Income“ u navigacijskom baru prikazuju se svi prihodi i informacije o njima, prikazano slikom 3.3.9. Nudi sve iste opcije kao i prikaz troškova uz različite kategorije.

Sl. 3.3.9 Prikaz prihoda

Kategorije prihoda su:

- Plaća
- Bonus
- Freelance
- Ulaganja
- Najam
- Prodaja
- Autorske naknade
- Mirovina
- Pokloni
- Ostalo

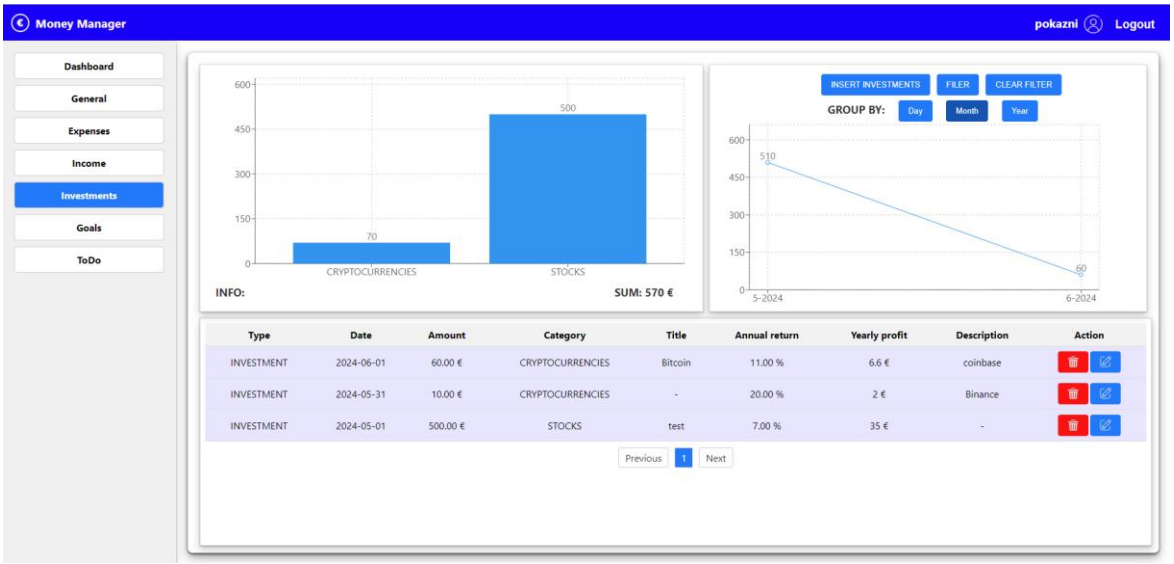
Investicije

Odabirom opcije „Investments“ u navigacijskom baru prikazuju se sve investicije i informacije o njima, prikazano slikom 3.3.10. Nudi sve iste opcije kao i kod prikaza troškova i prihoda uz različite kategorije te dodatan opcionalni parametar očekivani godišnji povrat. Moguće je filtriranje i sortiranje po vrijednostima očekivanog godišnjeg povrata, slika 3.3.11.

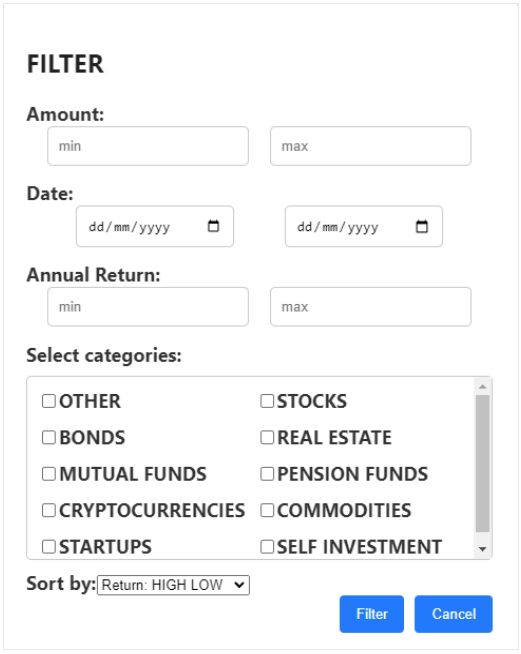
Kategorije investicija su:

- Dionice

- Obveznice
- Nekretnine
- Uzajamni fondovi
- Mirovinski fondovi
- Kriptovalute
- Roba
- Startup-ovi
- Osobna ulaganja
- Ostalo



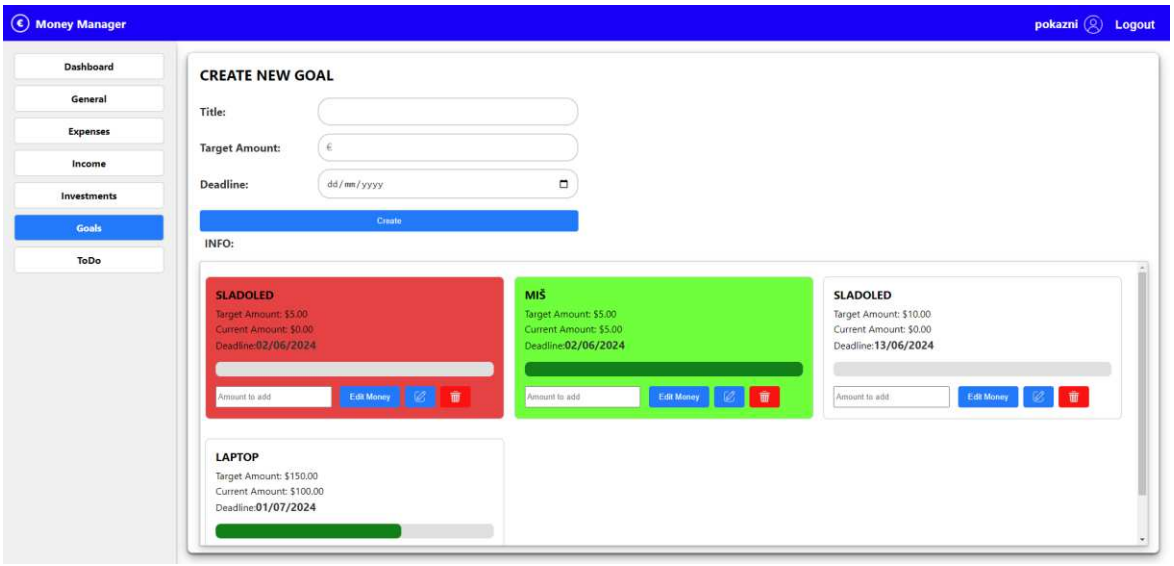
Sl. 3.3.10 Prikaz investicija



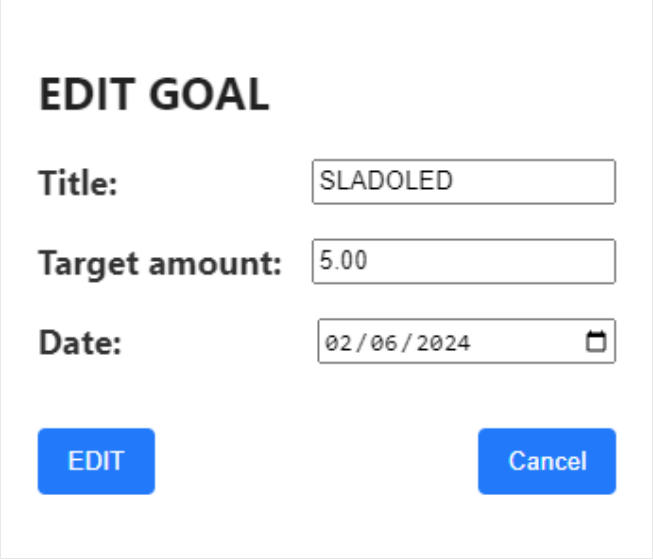
Sl. 3.3.11 Filter investicija

Ciljevi

Odabirom opcije „Goals“ u navigacijskom baru prikazuje se svi ciljevi, prikazano slikom 3.3.12. Ovaj odjeljak omogućava korisnicima postavljanje i praćenje financijskih ciljeva. Korisnici mogu definirati ciljeve, mijenjati pridijeljena sredstva i pratiti napredak. Moguće je urediti i izbrisati cilj. Pridijeljena sredstva se oduzimaju od ukupnog salda korisnika.



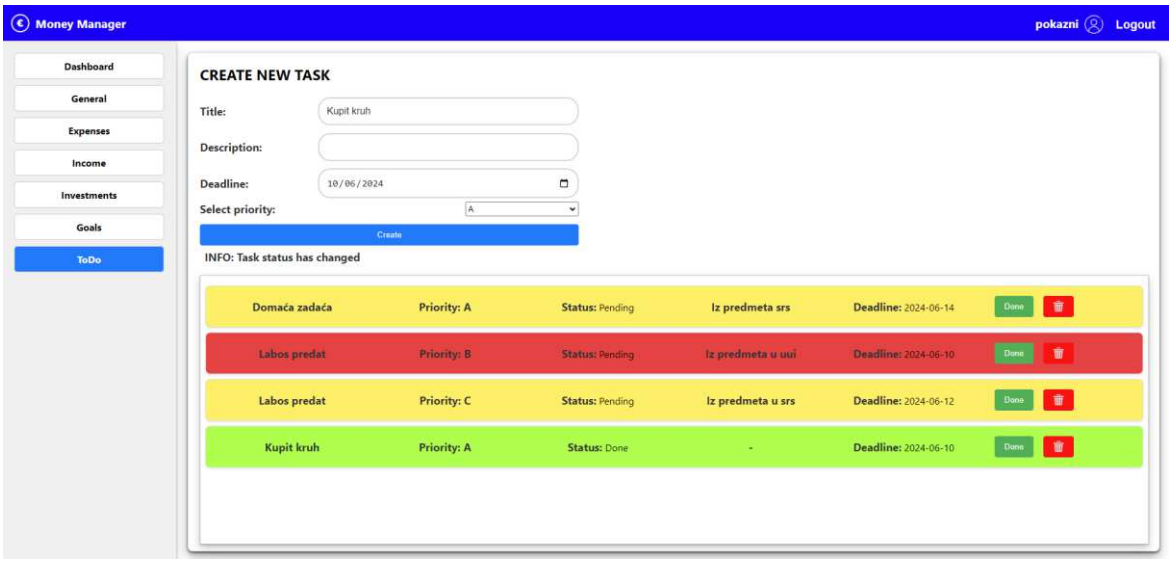
Sl. 3.3.12 Prikaz ciljeva



Sl. 3.3.13 Forma za uređivanje cilja

Zadaci

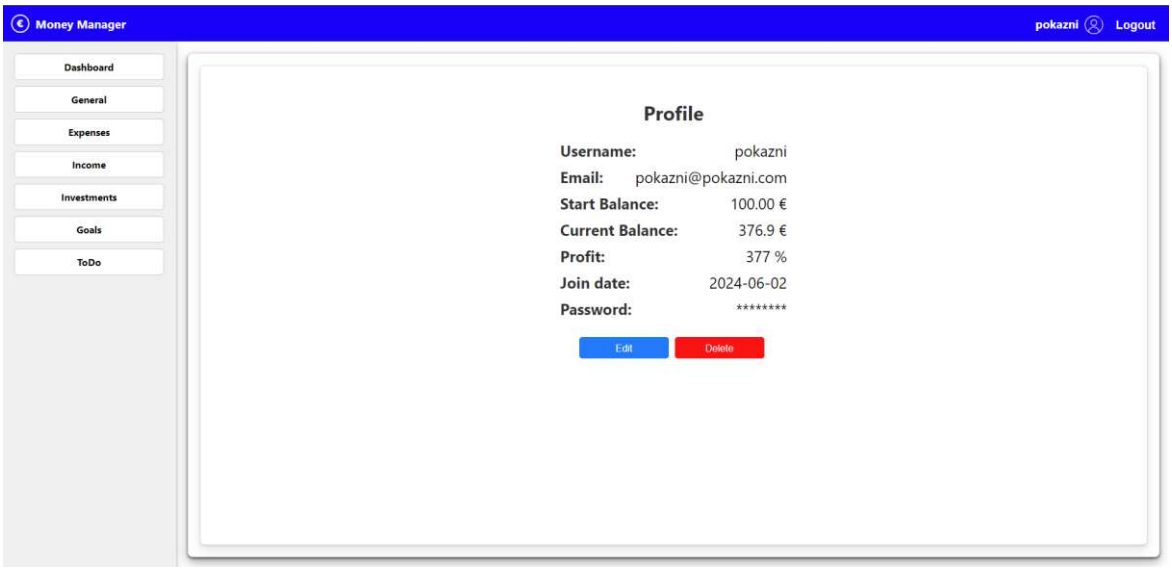
Odabirom opcije „ToDo“ u navigacijskom baru prikazuju se svi zadaci, prikazano slikom 3.3.14. Korisnici mogu kreirati zadatke unosom naslova, opisa (opcionalno), roka izvršenja i prioriteta te pratiti njihov status. Zadacima se klikom na gumb „Done“ može promijeniti status zadataka. Moguće je izbrisati zadatak klikom na crveni gumb pojedinog zadatka.



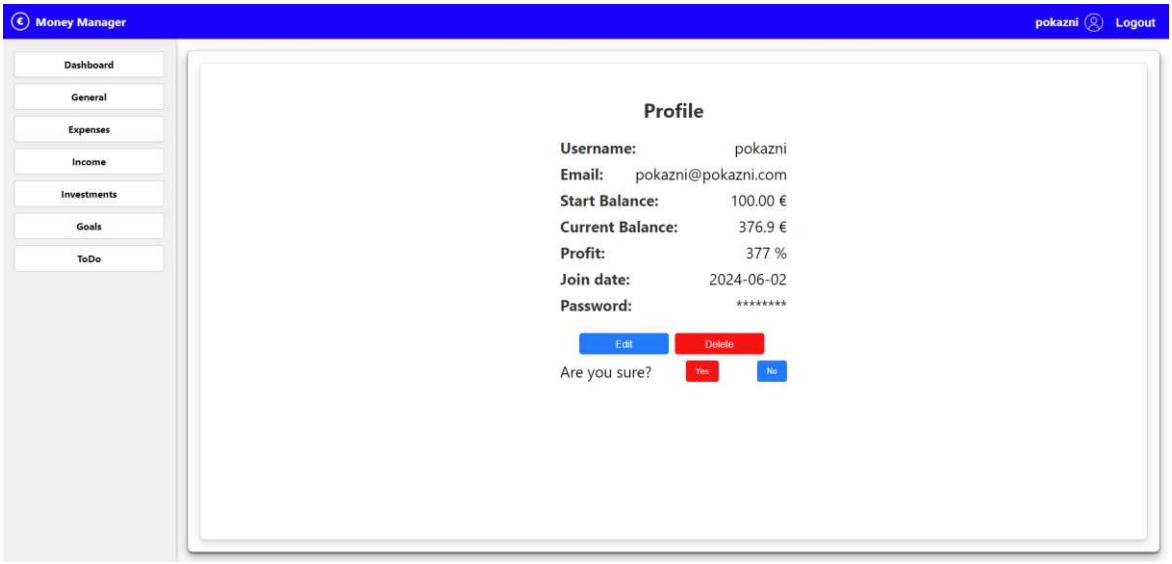
Sl. 3.3.14 Prikaz zadataka

Profil

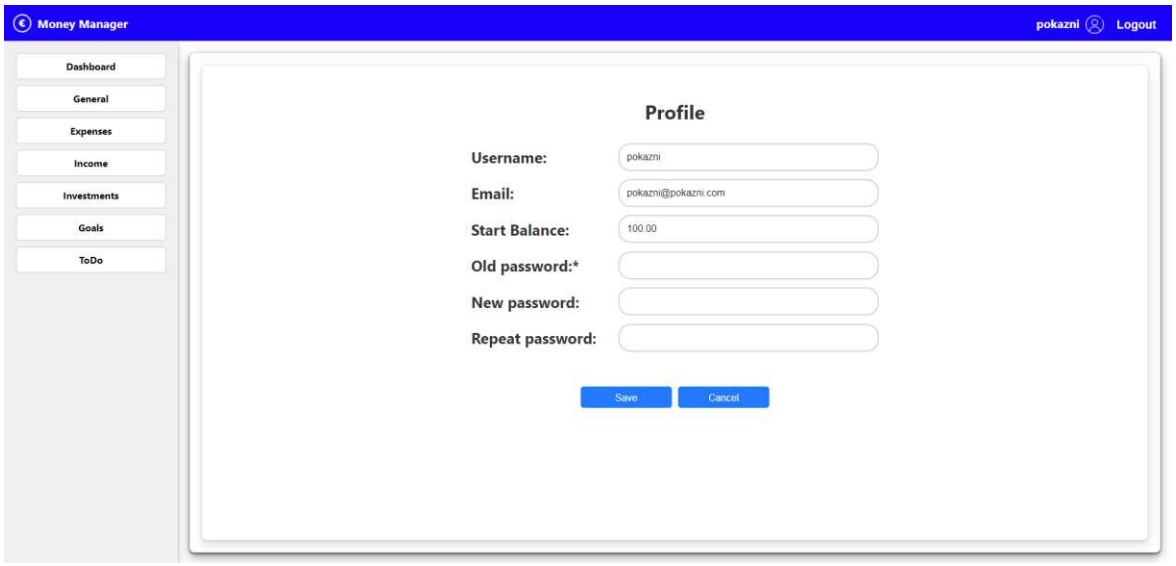
Klikom na korisničko ime ili na ikonu profila pristupamo profilu, prikazan slikom 3.3.15. Klikom na gumb „Edit“ otvara se forma za uređivanje informacija o računu, prikazan slikom 3.3.16. Klikom na gumb „Delete“ započinje se akcija trajnog brisanja cijelog računa sa svim podacima. Akciju treba potvrditi klikom na gumb „Yes“ dodatne provjere koja se prikaže, slika 3.3.17.



Sl. 3.3.15 Prikaz podataka o profilu



Sl. 3.3.16 Prikaz potvrde brisanja profila



Sl. 3.3.17 Forma za uređivanje podataka o profilu

Uz svaku promjenu korisničkih informacija obavezno je unijeti trenutnu lozinku. Klikom na gumb „Save“ se potvrđuje promjena. Ako dođe do greške, ispisuje se poruka o grešci, a polje koje je izazvalo grešku postaje crveno.

3.3.2. Administrator

Administratori imaju pristup dodatnim funkcionalnostima putem sučelja Django REST radnog okvira, prikazano na slici 3.3.18, kojem se pristupa „adresa_django_servera/admin“. Ovo sučelje omogućava sveobuhvatnu kontrolu nad podacima unutar sustava, uključujući korisničke podatke, transakcije, ciljeve i zadatke, slika 3.3.19.

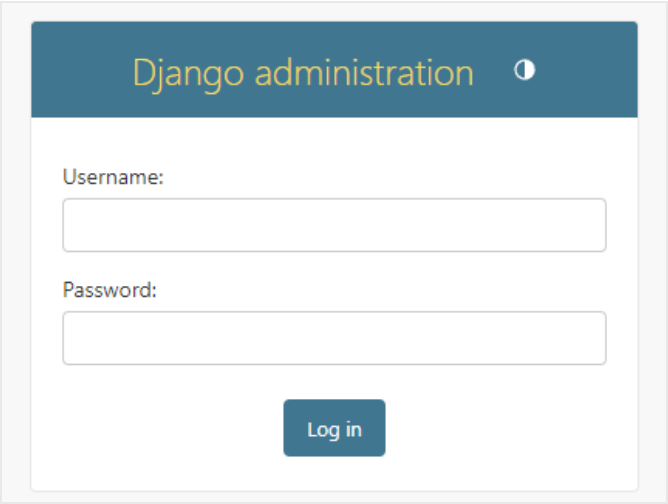
Uređivanje i brisanje podataka: Administratori mogu pregledavati, uređivati i brisati sve podatke u sustavu. To uključuje korisničke profile, financijske transakcije, postavljene ciljeve i zadatke. Primjer uređivanja podataka 3.3.20.

Kreiranje administratora

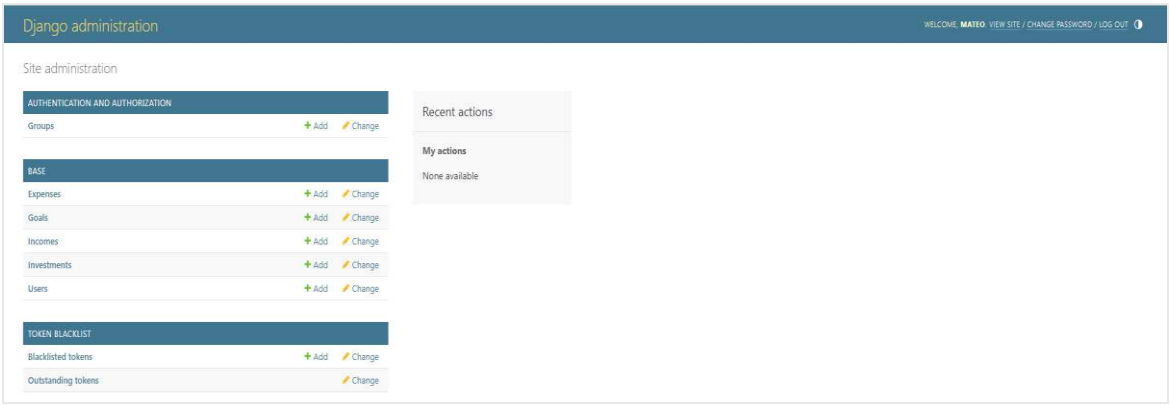
Kreiranje administratora je omogućeno putem konzole unutar korijenskog direktorija Django projekta. Naredba:

```
python manage.py createsuperuser
```

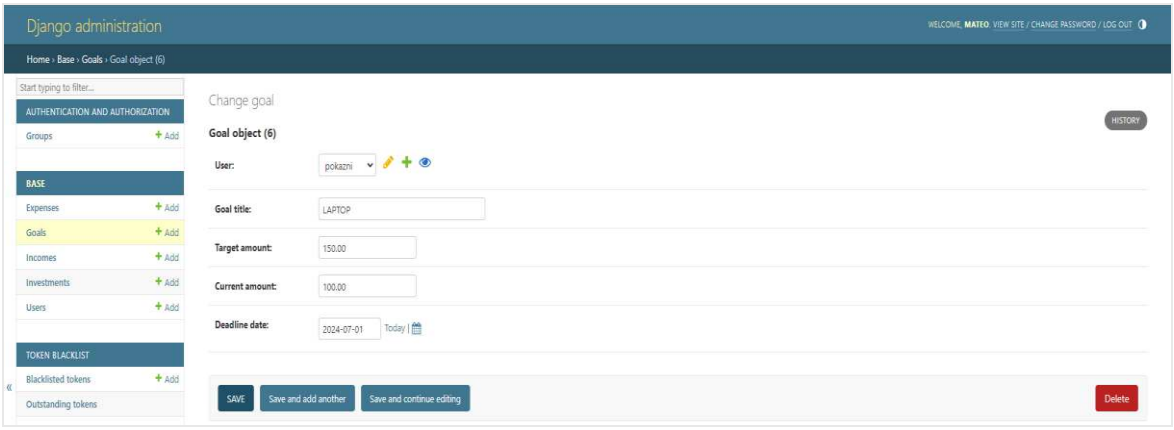
Nakon koje se popunjavaju informacije o administratoru kao što su ime, email i lozinka.



Sl. 3.3.18 Forma za prijavu administratora



Sl. 3.3.19 Administratorski prikaz svih podataka



Sl. 3.3.20 Uređivanje cilja kao administrator

Zaključak

Zadatak ovog završnog rada bio je napraviti bazu podataka i web aplikaciju za upravljanje osobnih financija. Nakon nekoliko mjeseci rada, to je i ostvareno. Implementirane su željene funkcionalnosti i priložena je detaljna dokumentacija.

Razvoj sustava se sastojao od nekoliko faza. Prvo je bilo proučavanje problema i zahtjeva korisnika, gdje smo identificirali ključne funkcionalnosti potrebne za učinkovito upravljanje financijskim podacima. Sljedeća faza uključivala je modeliranje i implementaciju baze podataka, čime smo osigurali sigurno i učinkovito pohranjivanje podataka. Zatim je uslijedio razvoj web aplikacije koja omogućuje jednostavno pregledavanje, dodavanje, izmjenjivanje i brisanje podataka. Konačno, sustav je prošao fazu testiranja i optimizacije kako bi se osigurala pouzdanost i performanse aplikacije.

Daljnji rad na ovom projektu bio bi usmjeren prema proširivanju sustava dodatnim funkcionalnostima kao što su naprednija analiza financijskih podataka, bolji alati za vizualizaciju statistika i grafikona te uvođenje dodatnih opcija za personalizaciju korisničkog sučelja. Još neka proširenja bi mogla biti vezana uz estetsko dotjerivanje korisničkog sučelja aplikacije ili optimiziranje i poboljšavanje tehničkih funkcionalnosti korištenjem naprednih JavaScript biblioteka.

Rad na ovom završnom radu se pokazao kao vrijedno i poučno iskustvo. Naučio sam raditi s tehnologijama kao što su SQL, React i Django, a u isto vrijeme sam proširio znanje o razvoju web aplikacija i upravljanju bazama podataka. Sva ta znanja će mi sigurno biti od koristi u daljnjem studiranju, ali i nakon njega.

Literatura

[1] Nastavni materijali iz kolegija Baze podataka, Fakultet elektrotehnike i računarstva, 2023.

[2] Nastavni materijali iz kolegija Razvoj programske potpore za web, Fakultet elektrotehnike i računarstva, 2023.

[3] W3Schools Online Web Tutorials, http://www.w3schools.com/, 15.05. 2024.

[4] Stack Overflow, http://www.stackoverflow.com, 10.05. 2024.

[5] Wikipedia, http://www.wikipedia.com/, 31.05. 2024.

[6] React Documentation, https://reactjs.org/docs/getting-started.html, 31.05. 2024.

[7] Django Documentation, https://docs.djangoproject.com/, 31.05.2024.

[8] Python Documentation, https://docs.python.org/, 31.05. 2024.

Sažetak

Baza podataka i web-aplikacija za praćenje osobnih financija

U ovom radu osmišljena je baza podataka i web aplikacija za upravljanje financijskim informacijama. U bazu podataka su pohranjeni podaci potrebni za praćenje, analizu i upravljanje financijama. Zatim je napravljena web-aplikacija koja omogućava pregled, unos, izmjenu i brisanje podataka. Kreirano je intuitivno korisničko sučelje koje korisnicima omogućuje jednostavno upravljanje njihovim financijskim podacima. Sustav obrađuje sve poslane podatke, provjerava njihovu ispravnost te poduzima akcije inicirane od strane korisnika. Baza podataka je napravljena korištenjem PostgreSQL-a. Web aplikacija je implementirana koristeći Django radni okvir na backendu i React na frontend-u. Korištene su tehnologije HTML5, CSS i JavaScript.

Ključne riječi: financijska aplikacija, Django, React, PostgreSQL, HTML5, CSS, JavaScript, upravljanje financijama

Summary

Database and Web Application for Managing Financial Information

This thesis presents the design of a database and a web application for managing financial information. The database stores the necessary data for tracking, analyzing, and managing finances. A web application was developed to enable users to view, add, edit, and delete data. An intuitive user interface was created to allow users to easily manage their financial information. The system processes all submitted data, verifies its accuracy, and takes actions initiated by the users. The database was built using PostgreSQL. The web application was implemented using the Django framework for the backend and React for the frontend. Technologies used include HTML5, CSS, and JavaScript.

Keywords: financial application, Django, React, PostgreSQL, HTML5, CSS, JavaScript, financial management