Upravljanje financijama s Personal Finance Manager web aplikacijom

Jurišić, Domagoj

VELEUČILIŠTE U VIROVITICI

Stručni prijediplomski studij Računarstvo

DOMAGOJ JURIŠIĆ

UPRAVLJANJE FINANCIJAMA S PERSONAL FINANCE MANAGER WEB APLIKACIJOM

ZAVRŠNI RAD

VIROVITICA, 2023 GODINA

VELEUČILIŠTE U VIROVITICI

Stručni prijediplomski studij Računarstvo

UPRAVLJANJE FINANCIJAMA S PERSONAL FINANCE MANAGER WEB APLIKACIJOM

ZAVRŠNI RAD

Predmet: Vjerojatnost i statistika

Mentor: Student:

Marijana Špoljarić, mag.educ.math.et inf., v.pred. Domagoj Jurišić

VIROVITICA, 2023 GODINA

Veleučilište u Virovitici

Stručni prijediplomski studij Računarstva - Smjer Programsko inženjerstvo

OBRAZAC 1b

ZADATAK ZAVRŠNOG RADA

Student/ica: **DOMAGOJ JURIŠIĆ** **JMBAG:** **0010223162**

Imenovani mentor: Marijana Špoljarić, mag. educ. math. et inf., v. pred.

Imenovani komentor: -

Naslov rada:

Upravljanje financijama s Personal Finance Manager web aplikacijom

Puni tekst zadatka završnog rada:

Teorijski dio:

- Opisati sve pojmove koji se koriste.
- Napisati sve statističke pojmove koji se koriste.
- Opisati cijelu aplikaciju.
-

Zadatak za izradu aplikacije:

- Personal Finance Manager je web aplikacija koja korisnicima omogućuje učinkovito upravljanje svojim financijama. Aplikacija je izgraditi na ReactJS-u za korisničko sučelje, Spring Boot Java za poslovnu logiku i MySQL bazu podataka za pohranu podataka.
- Aplikacija zahtjeva Login/Registraciju za ulaz. Postoje 4 Ekrana po kojima se korisnik može kretati: Home, Reports, Transaction Manager i Settings.
- Home je početna stranica aplikacije i prikazuje ulazne i izlazne troškove, kao i troškove po kategorijama. Postoje dvije kategorije: Prihod s računa (Income), Troškovi računa (Expenses) i Ulaganja, a svaka kategorija ima svoje podkategorije koje detaljnije prikazuju raspolaganje i trošenje novca.
- Reports treba sadržavati sve račune korisnika. Tu je tablica za navigaciju kroz račune s mogućnošću uređivanja i brisanja istih. Također se nalazi tražilica za pretraživanje tablice i funkcionalnost izvoza podataka u Excel.
- Transaction Management omogućava korisniku unos računa. Korisnik bira tip računa (Prihod ili Trošak), unosi cijenu, podkategoriju, datum (dan, mjesec, godina) te kratki opis računa.
- Settings služi za pregled korisničkih postavki. Na ovom ekranu korisnik može vidjeti osnovne podatke, promijeniti sliku ili šifru korisničkog računa, dodati novu podkategoriju, promijeniti valutu te obrisati korisnički račun.
- Prikaz dobitaka i rashoda po danu,

- Režije pretraživati koje su minimalne, koje su maksimalne, koliko u prosjeku mjesečno se izdvaja za režije, koliko je povećanje s obzirom na prošli mjesec
- Analiza potrošnje za namirnice, liječnika i sl.
- Ulaganja, koliko se mjesečno ulaže, mjesečni i godišnji prosjek, kolike su kamate, kolika je zarada dobit.
- Linearni trend kretanja ulaganja.
- Na istom linijskom gradu prikazati kretanje primitaga, ulaganja i troškova po danu.

Datum uručenja zadatka studentu/ici: 31.07.2023.
Rok za predaju gotovog rada: 08.09.2023.

Mentor:

Marijana Špoljarić, mag.educ.math. et inf., v.pred.

Dostaviti:
1. Studentu/ici
2. Povjerenstvu za završni rad - tajniku

UPRAVLJANJE FINANCIJAMA PERSONAL FINANCE MANAGER WEB APLIKACIJOM

Sažetak

U današnjem digitalnom dobu, upravljanje financijama postalo je ključno za svakodnevni život. Ovaj rad predstavlja web aplikaciju „Personal Finance Manager" koja omogućava korisnicima učinkovito upravljanje svojim financijama. Cilj rada bio je razviti aplikaciju koja je jednostavna za korištenje, ali pruža sveobuhvatne funkcionalnosti za upravljanje financijama. Aplikacije je izgrađena koristeći ReactJS za korisničko sučelje, Spring Boot Java za poslovnu logiku i MySQL bazu podataka za pohranu podataka. Rad detaljno opisuje svaku komponentu aplikacije, uključujući Home, Statistic, Reports, Transaction Manager i Settings stranice. Značajne funkcionalnosti poput upravljanja transakcijama, analize potrošnje i izvještajna također su istaknute. Ovaj rad pridonosi razumijevanju važnosti digitalnih alata u upravljanju osobnim financijama te predstavlja korak naprijed u pružanju rješenja za moderne financijske izazove.

Ključne riječi: *ReactJS, Spring Boot Java, MySQL,Prijavni Ekran, Registracijski Ekran, Navigacijska traka Dashboard, Statistika, Repots, Transaction Manager, Upravljanje financijama*

MANAGING FINANCES WITH THE PERSONAL FINANCE MANAGER WEB APPLICATION

Abstract

In today's digital age, managing finances has become crucial for everyday life. This paper introduces the web application "Personal Finance Manager" that allows users to effectively manage their finances. The aim of the work was to develop an application that is simple to use, yet offers comprehensive functionalities for financial management. The application was built using ReactJS for the user interface, Spring Boot Java for business logic, and MySQL database for data storage. The paper provides a detailed description of each component of the application, including the Home, Reports, Transaction Manager, and Settings pages. Significant features such as transaction management, spending analysis, and reporting are also highlighted. This paper contributes to the understanding of the importance of digital tools in managing personal finances and represents a step forward in providing solutions for modern financial challenges.

Keywords: *ReactJS, Spring Boot Java, MySQL, Login, Register, Navbar, Dashboard, Statistics, Repots, Transaction Manager, Financial management*

Sadržaj

1. Uvod

U svijetu tehnologija sve više dominira, a digitalizacija igra ključnu ulogu u gotovo svim aspektima našeg života Upravljanje financijama postaje sve složenije. Tradicionalne metode praćenja prihoda, rashoda i štednje često nisu dovoljno brze i prilagodljive za suvremene potrebe korisnika. Stoga je potrebna promjena u pristupu, gdje se kombiniraju najnovije tehnološke inovacije i tradicionalne financijske strategije kako bi se postigla optimalna ravnoteža između efikasnosti i sigurnosti. Upravljanje osobnim financijama nije samo zadatak računovođa ili financijskih stručnjaka. Svaka osoba treba imati alate i resurse koji će joj pomoći da bolje upravlja svojim financijama, te donosi informirane odluke. Moderna tehnologija pruža platformu za stvaranje takvih alata koji su prilagođeni potrebama svakog pojedinca. U ovom radu, predstavit će se web aplikaciju pod nazivom "Personal Finance Manager". Kroz ovaj rad, čitatelj će dobiti uvid u konceptualni dizajn, tehnologije korištene za razvoj, kao i glavne funkcionalnosti aplikacije. Osim toga, rad će se baviti važnošću upravljanja financijama u današnjem dobu, te kako aplikacija može pridonijeti boljoj financijskoj pismenosti i donošenju odluka. Također, rad će istražiti kako digitalni alati, poput "Personal Finance Manager" aplikacije, mogu transformirati način na koji ljudi pristupaju i upravljaju svojim financijama, čineći cijeli proces transparentnijim, efikasnijim i prilagodljivijim.

2. Pozadina i važnost financijskog upravljanja

Financijsko upravljanje odnosi se na aktivnosti i strategije pomoću kojih pojedinci ili organizacije upravljaju svojim novčanim sredstvima. To je jedan od najstarijih koncepta u ljudskoj povijesti, s korijenima koji sežu do drevnih civilizacija koje su koristile razne metode za praćenje svojih prihoda i rashoda [1].

2.1 Povijesna pozadina

Od prvih oblika trgovine, uključujući barter sustav – gdje su ljudi izravno razmjenjivali dobra i usluge bez korištenja novca – pa sve do uvođenja novčanih sustava, ljudi su tražili metode kako bi učinkovito upravljali svojim bogatstvom. Barter sustav, iako jednostavan, često je bio neučinkovit jer je zahtijevao dvostruku želju – oba sudionika morala su imati nešto što druga strana želi. S pojavom pisane riječi drevne civilizacije, kao što su Babilonci i Egipćani, počeli su voditi knjige o svojim financijskim transakcijama. Razvojem tržišne ekonomije i bankarskog sustava, potreba za sofisticiranim metodama financijskog upravljanja postala je sve izraženija [1].

2.2 Moderno financijsko upravljanje

Danas, u doba globalizacije i digitalizacije, financijsko upravljanje je postalo kompleksnije nego ikad prije. Globalna tržišta, kompleksni financijski instrumenti i brza razmjena informacija zahtijevaju da pojedinci i organizacije budu opremljeni znanjem i alatima za učinkovito upravljanje svojim financijskim resursima [1].

2.3 Važnost financijskog upravljanja

Upravljanje financijama igra ključnu ulogu u osiguranju stabilnosti i održivosti, bilo da se radi o pojedincu, obitelji ili korporaciji. Evo nekoliko razloga zbog kojih je financijsko upravljanje od vitalne važnosti:

-Budžetiranje i planiranje: Upravljanje financijama pomaže u postavljanju jasnih ciljeva, definiranju prioriteta i usmjeravanju resursa prema postizanju tih ciljeva.

-Sprečavanje dugova: Praćenje prihoda i rashoda osigurava da se živi unutar svojih sredstava, smanjujući potrebu za zaduživanjem.

-Osiguranje budućnosti: Pravilno upravljanje financijama pomaže u štednji i investiranju za budućnost, bilo da je riječ o mirovini, obrazovanju djece ili kupnji nekretnine.

-Odlučivanje na temelju informacija: S pravilnim alatima i znanjem, pojedinci mogu donositi informirane odluke koje će im omogućiti ostvarivanje dugoročnih ciljeva.

U doba digitalizacije, alati kao što je "Personal Finance Manager" pružaju priliku za bolje, preciznije i transparentnije upravljanje financijama, što je od suštinske važnosti u brzom i dinamičnom svijetu u kojem živimo.

3. Tehnologije korištene u izradi aplikacije

U izradi "Personal Finance Manager" web aplikacije, korištene su različite suvremene tehnologije kako bi se osigurale najbolje funkcionalnosti i performanse. Svaka od ovih tehnologija ima svoju specifičnu ulogu i doprinos u izradi aplikacije.

3.1 ReactJS

ReactJS ili češće samo „React“, jedna je od vodećih JavaScrpit biblioteka specijalizirana za izgradnju korisničkih sučelja. Nastao je kao rezultat inovativnih inicijativa u Facebooku i od tada se široko usvojio u industriji web razvoja [2].

React promovira modularni pristup razvoja aplikacija. Aplikacija je sastavljana od pojedinačnih komponenti, svaka sa svojom funkcionalnošću i sučeljem. Ovo omogućava ponovnu upotrebu koda i čini aplikacije lako održivim [2].

Isječak programskog koda 1: Primjer pozivanja komponente.

```
1. import React, { useState } from "react";
2. import { Container, Card, Row, Col, Button } from "react-bootstrap";
3. import Navbar from "../Components/Navigation/Navbar";
4. import Balance from "../Components/Balance/Balance";
5. import IncomeExpensesGraph from "../Components/Charts/IncomeExpensesGraph";
6. import IncomeExpensesCard from "../Components/Charts/IncomeExpensesCards";
7. import "../Styles/Home.css";
8. function Home() {
9.   const [isIncomeExpensesCardVisible, setIncomeExpensesCardVisible] = useState(false);
10.  return (
11.      <Navbar />
12.         <Container>
13.              <Row className="mt-4">
14.                <Col>
15.                   <p className="lead">Gain insights into your financial activity.</p>
16.                </Col>
17.               </Row>
18.               <Row className="mt-4">
19.                 <Col>
20.                <Card>
21.                   <Row>
22.                         <Col>
23.                           <IncomeExpensesGraph />
24.                          </Col>
25.                        </Row>
26.                    </Card>
```

```
                    </Col>
                </Row>
            </Container>
    );
}
export default Home;

```

React koristi koncept Virtualnog DOM-a što značajno povećava performanse aplikacije. Umjesto da izravno ažurira stvarni DOM svaki put kada dođe do promjene. React prvo izrađuje „snapshot“ trenutnog DOM-a, ažurira ga, a zatim ga uspoređuje sa stvarnim DOM-om, ažurirajući samo one dijelove koji su se promijenili. React koristi JSX, kombinaciju JavaScripta i XML-a, koja omogućuje jednostavnije intuitivnije definiranje korisničkih sučelja unutar JavaScripta.

3.1.1 Primjena ReactJS-a u „Personal Finance Manager“ aplikaciji

U kontekstu „Personal Finance Manager“ aplikacije, ReactJS je odigrao ključnu ulogu kao osnova za izradu front-end dijela aplikacije.

-Modularnost i ponovna upotreba koda: komponentna arhitektura React-a korištena je za izgradnju različitih segmenata korisničkog sučelja, uključujući forme, liste, grafikone. Ovaj pristup omogućava ponovnu upotrebu komponenti na različitim dijelovima aplikacije, čime se smanjuje redundancija koda i povećava efikasnost razvoja.

-Dinamičnost i interaktivnost: korištenjem React-a, aplikacije je obogaćena dinamičnim značajkama poput integrativnih grafova i formi koje se ažuriraju u stvarnom vremenu bez potrebe za osvježivanjem stranice.

3.1.2 Komunikacija s API-jem pomoću axios-a

U suvremenom web razvoju, ispravna komunikacija s API-jem postala je nezaobilazna. Kako aplikacije postaju sve kompleksnije, potreba za robusnim, ali jednostavnim rješenjima za pristup podacima raste eksponencijalno. U ovom odjeljku razmotrit ćemo kako aplikacija "Personal Finance Manager" koristi biblioteku axios za ostvarivanje te ključne komunikacije. „Income Service“ služi kao centralna točka za pristup podacima o prihodima. Evo nekoliko karakteristika i prednosti upotrebe axios-a unutar ove servisne klase:

-Asinkrona priroda: U okviru modernog web razvoja prepoznate su prednosti asinkronog programiranja. Axios omogućuje integraciju te prednosti kroz „async/await“ sintaksu. U praksi, kada „IncomeService“ inicira zahtjev prema API-ju, ostatak aplikacije nije blokiran. Aplikacija može nastaviti s drugim operacijama dok čeka odgovor.

-Personalizacija zahtjeva: Svaki korisnik aplikacije ima jedinstvene podatke. Da bi se osigurala ispravna dostava korisnički specifičnih podataka, axios koristi dinamične URL-ove, konkretno, dodavanjem korisnikovog ID-a unutar zahtjeva. Ova tehnika garantira da se pravim korisnicima dostavljaju odgovarajući podaci.

Isječak programskog koda 2: Primjer pozivanja „IncomeService“ servisa unutar komponente.

```
1. import React, { useState, useEffect } from "react";
2. import { Container, Card, Row, Col } from "react-bootstrap";
3. import { Line } from "react-chartjs-2";
4. import IncomeService from "../../API/IncomeService";
5. import ExpensesService from "../../API/ExpensesService";
6. import CategoryFilter from "../CategoryFilter.js";
7. function IncomeExpensesGraph() {
8.   const [incomeData, setIncomeData] = useState([]);
9.   const [expensesData, setExpensesData] = useState([]);
10.  const [selectedCategories, setSelectedCategories] = useState([]);
11.  useEffect(() => {
12.    async function fetchData() {
13.      try {
14.        const incomeResponse = await IncomeService.getIncomeData();
15.        const expensesResponse = await ExpensesService.getExpensesData();
16.        incomeResponse.sort((a, b) => new Date(a.date) - new Date(b.date));
17.        expensesResponse.sort((a, b) => new Date(a.date) - new Date(b.date));
18.        setIncomeData(incomeResponse);
19.        setExpensesData(expensesResponse);
20.      } catch (error) {
21.        console.error("Error fetching data:", error);
22.      }
23.    }
24.    fetchData();
25.  }, []);
26.
```

Isječak programskog koda 3: Primjer pozivanja API-ja iz servisa „IncomeService“.

```
1. import axios from "axios";
2. import user from "../Autentication/User/User";
3.
4. const apiUrl = "http://localhost:8080";
5.
6. const IncomeService = {
7.   getIncomeData: async () => {
8.     try {
9.       const response = await axios.get(`${apiUrl}/income/${user.getUserId()}`);
10.      return response.data;
11.    } catch (error) {
12.      throw error;
13.    }
14.  },
15.  getIncomeTrendData: async () => {
16.    try {
17.      const response = await axios.get(
18.        `${apiUrl}/income/trend/${user.getUserId()}`
19.      );
20.      return response.data;
21.    } catch (error) {
22.      throw error;
23.    }
24.  },
25. };
26.
27. export default IncomeService;
28.
```

3.2 Spring Boot Java

Programski okvir pod nazivom Spring Boot, nadogradnja je Spring programske platforme koja se u današnjem razvijenom svijetu globalnih tehnologija uvelike koristi kao baza za razvoj mobilnih i transakcijskih aplikacija. Obzirom na pojednostavljenu konfiguraciju prilikom samog razvijanja aplikacije, spomenuti programski okvir ujedno i ubrzava postupak implementacije, odnosno samo korištenje koje se prilagođava krajnjem korisniku. Obzirom na njegove sposobnosti da pojednostavni konfiguraciju i ubrza postupak izrade aplikacija, postao je gotovo standard za mnoge Java razvojne projekte, uključujući "Personal Finance Manager" aplikaciju.

3.2.1 Primjena Spring Boot-a u „Personal Finance Manager“ aplikaciji

U kontekstu "Personal Finance Manager" aplikacije, Spring Boot je poslužio kao temelj za izgradnju poslovne logike i obrade podataka.

-RESTful web servisi: Pomoću „spring-boot-starter-web“, aplikacija je dizajnirana da pruži RESTful servise, omogućujući front-endu asinkroni pristup podacima i funkcionalnostima poslovne logike.

-Integracija s bazom podataka: Korištenje „spring-boot-starter-data-jpa“ omogućilo je fluidnu integraciju s bazom podataka koristeći Java Persistence API (JPA). Ovo je pojednostavilo CRUD operacije, transformaciju objekata i upravljanje transakcijama [3]. Da bismo bolje razumjeli ovu integraciju, evo primjera kako aplikacija upravlja podacima o prihodima (income) preko programskog isječka 4:

Isječak programskog koda 4: Model(Entitet): „Income“ predstavlja prihod u našem sustavu

```
1. @Entity
2. public class Income {
3.     @Id
4.     @GeneratedValue(strategy = GenerationType.IDENTITY)
5.     private Long id;
6.
7.     private Double amount;
8.     @ManyToOne
9.     @JoinColumn(name = "income_category_id")
10.    private IncomeCategory category;
11.    private String date;
12.    private String notes;
13.    @ManyToOne
14.    @JoinColumn(name = "user_id")
15.    private User user;
16.    public Income() {
17.    }
18.    public Income(Long id, Double amount, IncomeCategory category, String date, String
notes, User user) {
19.
20.        this.id = id;
21.        this.amount = amount;
22.        this.category = category;
23.        this.date = date;
24.        this.notes = notes;
25.        this.user = user;
26.    }
```

Isječak programskog koda 5: Repozitorij: „IncomeRepository“ omogućuje izvođenje CRUD operacija nad prihodima

```
1. @Repository
2. public interface IncomeRepository extends JpaRepository<Income, Long> {
3.     List<Income> findByUser(User user);
4.
5.     @Query("SELECT i FROM Income i WHERE i.user = :user AND i.category = :category")
6.     List<Income> findByUserAndCategory(User user, IncomeCategory category);
7.
8.     @Modifying
9.     @Query("DELETE FROM Income i WHERE i.user = :user AND i.id = :incomeId")
10.    void deleteByUserAndIncomeId(@Param("user") User user, @Param("incomeId") Long
incomeId);
11.
12.
13.    @Query("SELECT i FROM Income i WHERE i.user = :user AND i.date = :date")
14.    List<Income> findByUserAndDate(@Param("user") User user, @Param("date") String date);
15.
16.    @Query("SELECT i FROM Income i WHERE i.user = :user ORDER BY i.date ASC")
17.    List<Income> findAllIncomesByUser(@Param("user") User user);
18.
19.    boolean existsByCategory(IncomeCategory category);
20. }
21.
```

Isječak programskog koda 6: Kontroler: „IncomeController“ upravlja dolaznim HTTP zahtjevima vezanim za prihode.

```
1. @RestController
2. @CrossOrigin("http://localhost:3000")
3. public class IncomeController {
4.
5.     private final IncomeRepository incomeRepository;
6.     private final IncomeCategoryRepository incomeCategoryRepository;
7.     private final UserRepository userRepository;
8.
9.     @Autowired
10.    public IncomeController(IncomeRepository incomeRepository, IncomeCategoryRepository
incomeCategoryRepository, UserRepository userRepository) {
11.        this.incomeRepository = incomeRepository;
12.        this.incomeCategoryRepository = incomeCategoryRepository;
13.        this.userRepository = userRepository;
14.    }
15.
16.    @PostMapping("/income")
```

```
17.     String addIncome(@RequestBody IncomeData incomeData) {
18.         Income income = new Income();
19.         income.setAmount(incomeData.getAmount());
20.         income.setDate(incomeData.getDate());
21.         income.setNotes(incomeData.getNotes());
22.
23.         IncomeCategory incomeCategory =
incomeCategoryRepository.findByCategory(incomeData.getCategory())
24.                 .orElseThrow(() -> new IllegalArgumentException("Invalid income
category"));
25.
26.         income.setCategory(incomeCategory);
27.         User user = userRepository.findById(incomeData.getUserId())
28.                 .orElseThrow(() -> new IllegalArgumentException("Invalid user ID"));
29.
30.         income.setUser(user);
31.         incomeRepository.save(income);
32.
33.         return "Income added successfully";
34.
```

-Sigurnost: Spring Boot, u kombinaciji sa Spring Security, pruža slojeve zaštite, autentifikaciju i autorizaciju, osiguravajući da podaci korisnika i aplikacije ostanu sigurni [3].

Zaključno, u "Personal Finance Manager" aplikaciji, Spring Boot je omogućio brzi razvoj, visoku efikasnost i integraciju s nizom drugih tehnologija i alata. Njegova fleksibilnost i širina mogućnosti čine ga idealnim izborom za suvremene web aplikacije.

3.3 MySQL Wokrbench

MySQL Workbench predstavlja suvremenu platformu za upravljanje različitim aspektima rada s MySQL bazama podataka. Svojim sveobuhvatnim alatima, omogućava razvojnicima i administratorima baza da bez problema kreiraju, testiraju, održavaju i optimiziraju svoje baze podataka [4].

3.3.1 Osnovne karakteristike MySQL Workbench-a

-Vizualni dizajn: Omogućava kreiranje entitetsko-relacijskih dijagrama (ERD) koji pomažu u vizualnom modeliranju baze podataka. Korisnici mogu lako dodavati, mijenjati ili brisati tablice, odnose, indekse i druge objekte baze podataka.

-SQL razvoj: Pruža integrirani SQL urednik s mogućnošću istraživanja i optimizacije upita, što značajno olakšava razvoj i testiranje SQL koda.

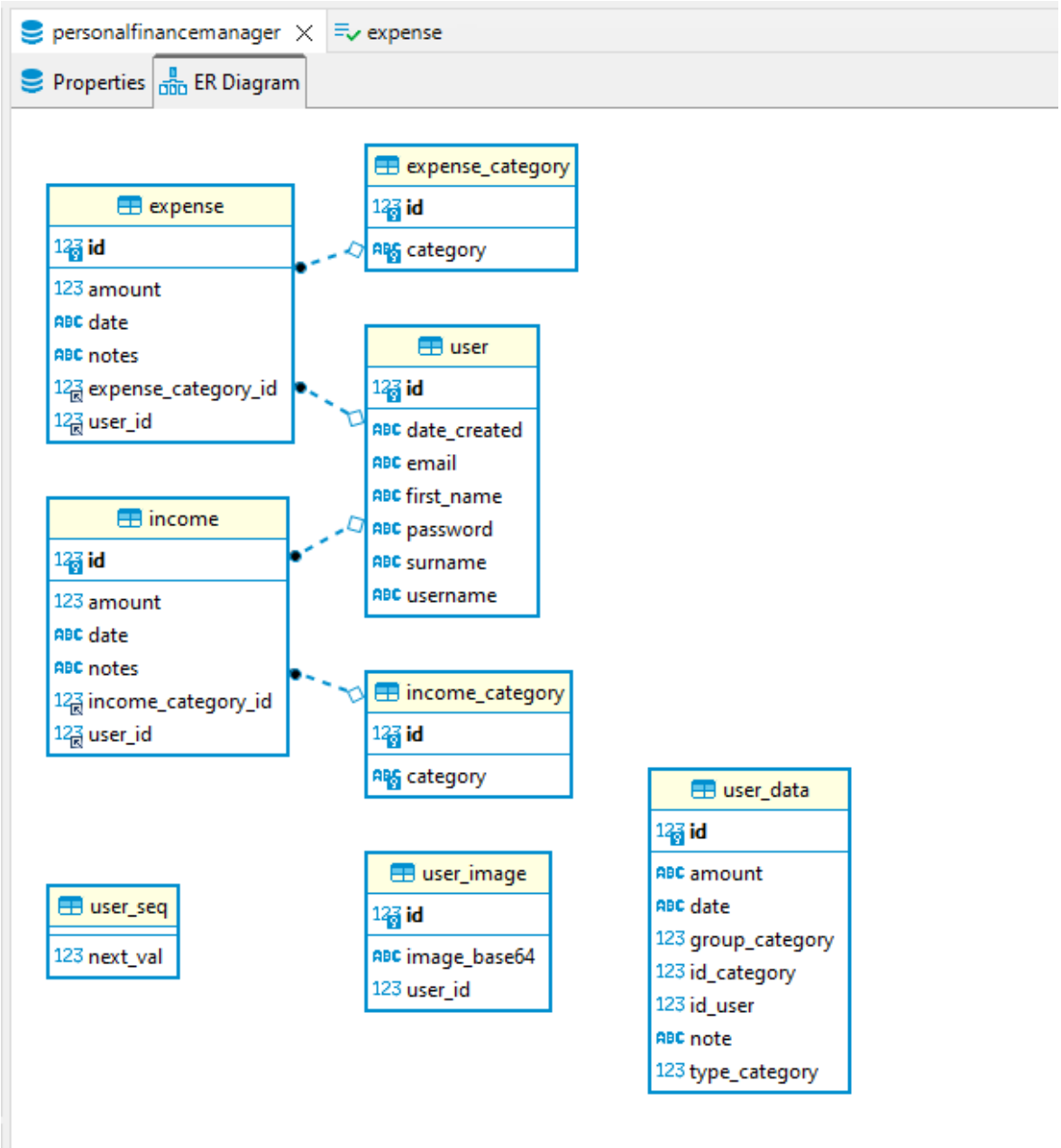
-Administracija: Sadrži niz alata za upravljanje MySQL serverima, uključujući konfiguraciju, upravljanje korisnicima, nadzor resursa i performansi te backup i oporavak podataka [5].

-Optimizacija i analiza: Ugrađeni alati omogućuju prepoznavanje i optimizaciju sporih upita, analizu opterećenja baze podataka te pružaju preporuke za poboljšanje performansi [5].

3.3.2 Primjena MySQL Workbench-a u „Personal Finance Manager“ aplikaciji

-Dizajniranje baze podataka: U početnoj fazi razvoja, MySQL Workbench korišten je za kreiranje entitetsko-relacijskog dijagrama aplikacije. Kroz ovu vizualnu reprezentaciju, jasno je definirana strukturu podataka, odnose između entiteta. Svi SQL upiti koji su korišteni unutar aplikacije prvotno su testirani unutar integriranog SQL urednika. Ovo je osiguralo točnost i performanse svakog upita prije njegove integracije u Spring Boot aplikaciju.

Integracija s Spring Boot-om kroz mysql-connector-j, MySQL Workbench baza podataka integrirana je s backend dijelom aplikacije, omogućavajući fluidnu komunikaciju između aplikacijske logike i podataka. MySQL Workbench predstavljao je nezamjenjiv alat u procesu razvoja "Personal Finance Manager" aplikacije. Pružio je robustnu platformu za rad s bazom podataka, dok je njegova integracija s ostalim tehnologijama osigurala da je aplikacija efikasna, skalabilna i sigurna.



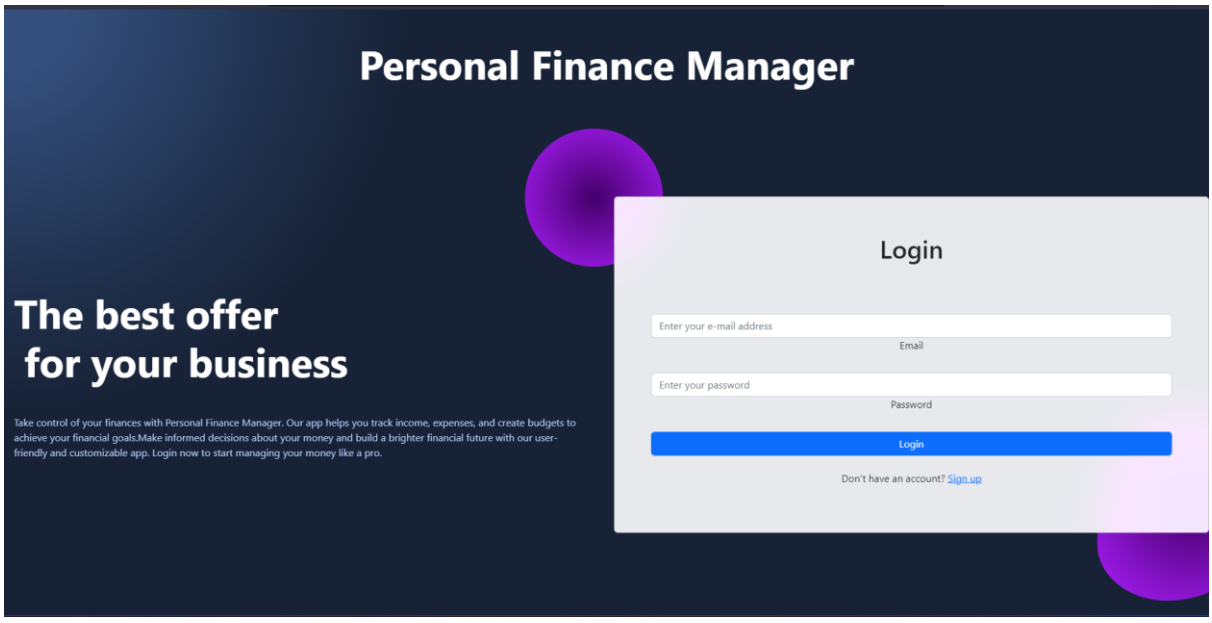
Slika 1.ER dijagram shema baze

4. Detaljan opis aplikacije

U prethodnom poglavlju dotakli smo se osnovnih principa i tehnologija korištenih u izradi „Personal Finance Manager“ aplikacije. U ovom poglavlju, detaljno je opisana aplikacija, njeni ključni elementi, te funkcionalnosti.

4.1 Prijavni Ekran (Login)

Prijavni ekran omogućava korisnicima pristup osobnim financijskim podacima putem e-mail-a i lozinke. Jedan od ključnih vizualnih elemenata ovog ekrana je naslov aplikacije "Personal Finance Manager", koji je jasno istaknut. Ovim naslovom korisnik odmah postaje svjestan aplikacije koju koristi. Ispod naslova, korisnik može pročitati uvodni tekst koji pruža kratak i precizan uvid u svrhu aplikacije. Tekst ističe prednosti korištenja aplikacije za upravljanje financijama. Centralni dio ekrana zauzima formular za prijavu. U njemu se nalaze dva glavna polja: polje za e-mail, gdje korisnici unose svoju e-mail adresu, i polje za lozinku, gdje korisnici unose svoju lozinku. Pored tih polja, postoji i dugme za prijavu. Kada korisnici pritisnu to dugme, aplikacija pokušava provjeriti identitet korisnika. Ako postoji problem s podacima za prijavu ili neka druga greška, korisnicima se prikazuje odgovarajuća poruka o grešci. Na dnu formulara nalazi se poveznica koja vodi korisnike na registracijski ekran, ako još uvijek nemaju račun u ovoj aplikaciji.



Slika 2. Prijavni ekran aplikacije

4.2 Registracijski Ekran (Register)

Registracijski ekran omogućava novim korisnicima kreiranje korisničkog profila unutar aplikacije. Kako bi to postigli, korisnici trebaju pružiti određene informacije poput svoje e-mail adrese, lozinke i drugih bitnih podataka. Važno je napomenuti da ovaj opis pruža opći pregled registracijskog ekrana temeljen na standardnim funkcionalnostima. Kada korisnik otvori stranicu za registraciju, primijetiti će naslov, označen kao "Registracija". Taj naslov jasno ukazuje da se korisnik nalazi na pravom mjestu kako bi kreirao svoj račun. Središnji dio ekrana zauzima formular za registraciju. U ovom formularu, prvo polje je obično namijenjeno unosu e-mail adrese. Zatim slijedi polje gdje korisnici kreiraju i unose željenu lozinku. Formular sadržava i dodatna polja imena, prezimena i sličnih informacija. Nakon što korisnici ispune sva potrebna polja, mogu kliknuti na dugme za registraciju kako bi završili cijeli postupak. Za one koji već imaju račun, na dnu formulara nalazi se poveznica koja ih preusmjerava natrag na prijavni ekran.

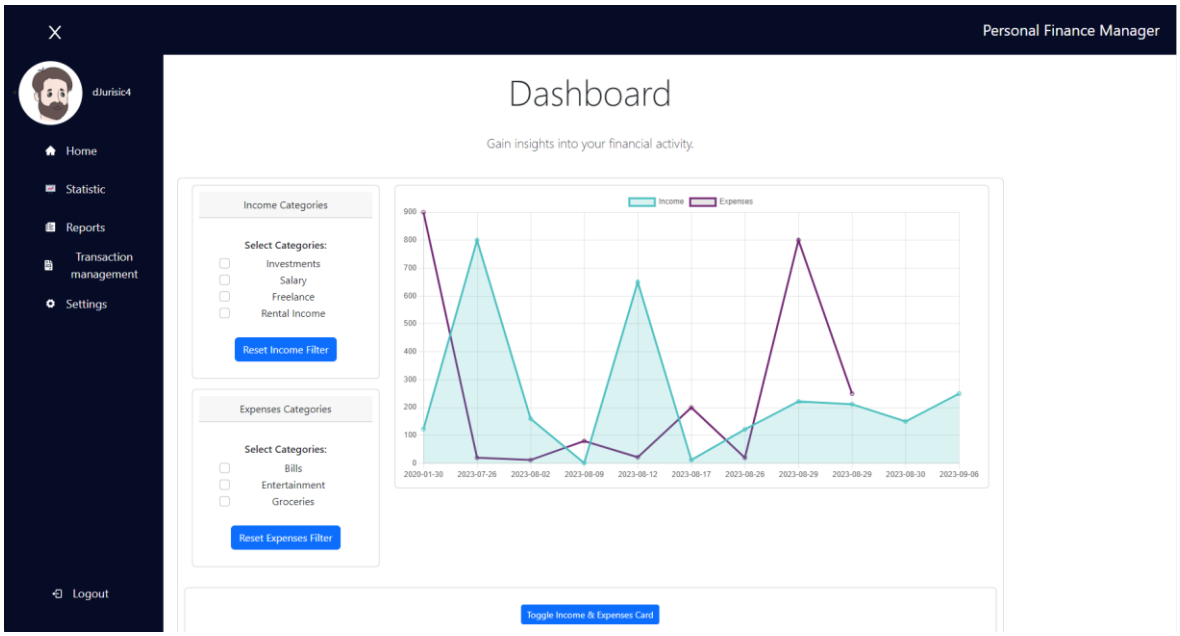
Slika 3. Registracijski ekran aplikacije

4.3 Navigacijska Traka (Navbar)

Navigacijska traka predstavlja ključni alat unutar aplikacije, osiguravajući korisnicima mogućnost jednostavnog i brzog prebacivanja između različitih dijelova aplikacije. Ova komponenta sadrži niz ikona koje služe kao prečaci za određene segmente aplikacije. Na primjer, ikona kuće obično vodi korisnika natrag na početnu stranicu, dok

druge ikone, poput onih za statistiku ili izvještaje, omogućuju pristup specifičnim funkcionalnostima i podacima. Jedan od važnih dijelova navigacijske trake je korisnički profil. U ovom segmentu, korisnik može vidjeti svoju profilnu sliku i korisničko ime. Za optimalnu iskustvenu fleksibilnost, aplikacija nudi mogućnost prilagodbe vidljivosti navigacijske trake. Korisnici imaju opciju proširivanja ili skrivanja cijele trake s pomoću određenih ikona. Na primjer, korištenjem ikone „FaBars“, traka se može prikazati u punom obliku, dok ikona „AiOutlineClose“ omogućava njezino zatvaranje, čime korisnik može maksimizirati prostor za rad na glavnom dijelu aplikacije. Odjava: Ova funkcija omogućava korisnicima odjavu iz sustava.

Slika 4. Prikaz zatvorene navigacijske trake

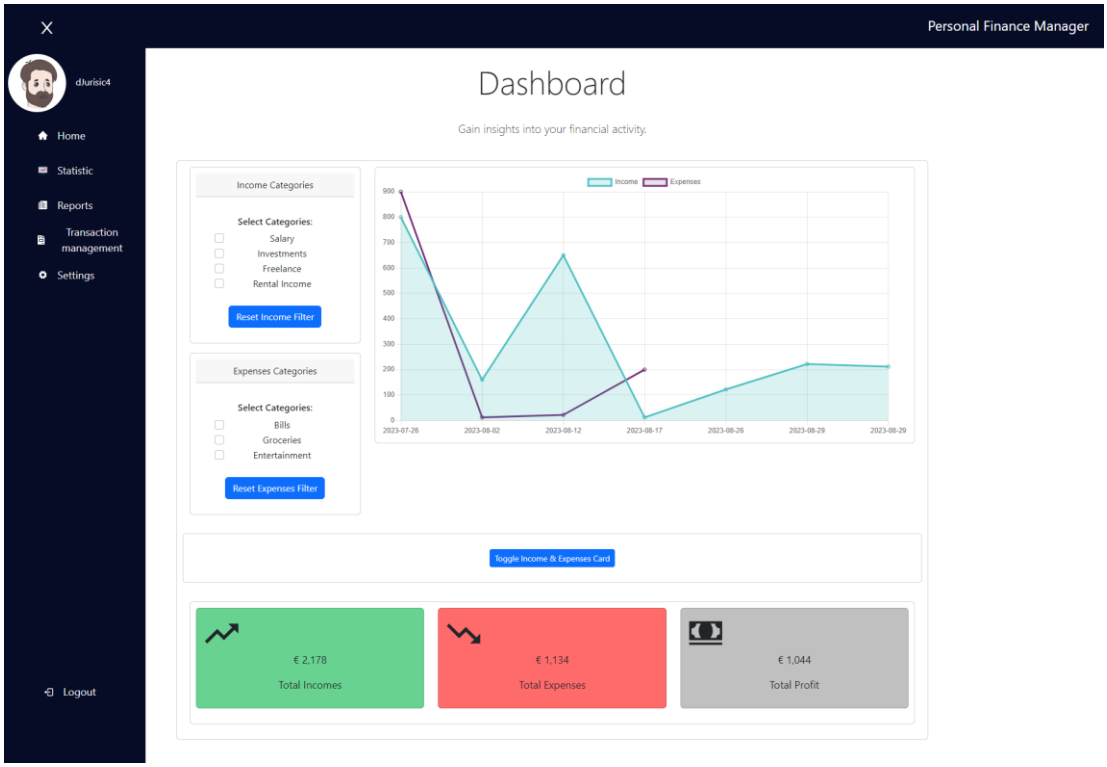


Slika 5. Prikaz rastvorene navigacijske trake

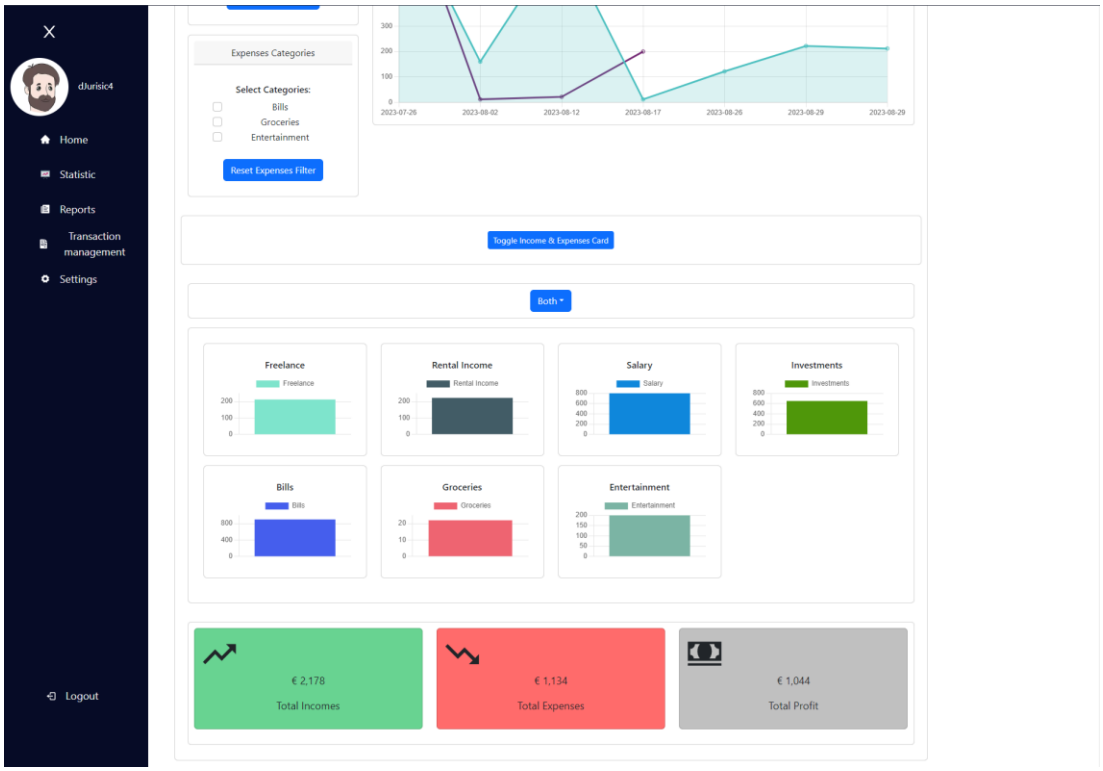
4.4 Početna Stranica (Home)

Početna stranica, nazivana „Dashboard“, je srce aplikacije, prikazuje sve korisnikove financijskih aktivnosti na jednom mjestu. Na vrhu ovog ekrana, jasan naslov informira korisnika da se nalazi na svojoj nadzornoj ploči, poslužujući kao orijentir unutar aplikacije.Korisnicima je na raspolaganju i uvodna poruka koja glasi: "Dobijte uvid u svoju

financijsku aktivnost." Ova rečenica ne samo da pojašnjava svrhu stranice, već i motivira korisnika da aktivno prati i analizira svoje financijske podatke.Jedna od centralnih komponenti početne stranice je „IncomeExpensesGraph“. Ovaj interaktivni grafikon nudi korisnicima mogućnost da na vizualan način prate dinamiku svojih prihoda i rashoda tijekom vremena. S obzirom na njegovu važnost, dizajniran je tako da bude intuitivan i lako čitljiv. Uz to, tu je “IncomeExpensesCard“, kartica koja korisnicima pruža detaljan uvid u specifične informacije o njihovim prihodima i rashodima. Ako korisnik želi pojednostaviti pregled svoje nadzorne ploče, postoji mogućnost skrivanja ove kartice korištenjem gumba "Prekidač prihoda i rashoda". Za one koji traže sažeti prikaz svojih financija, bilanca na početnoj stranici kombinira sve prihode i rashode, pružajući korisniku jasan uvid u njegovo trenutačno financijsko stanje.



Slika 6. Home/Dashboard ekran aplikacije



Slika 7. Prikaz kartica kategorija

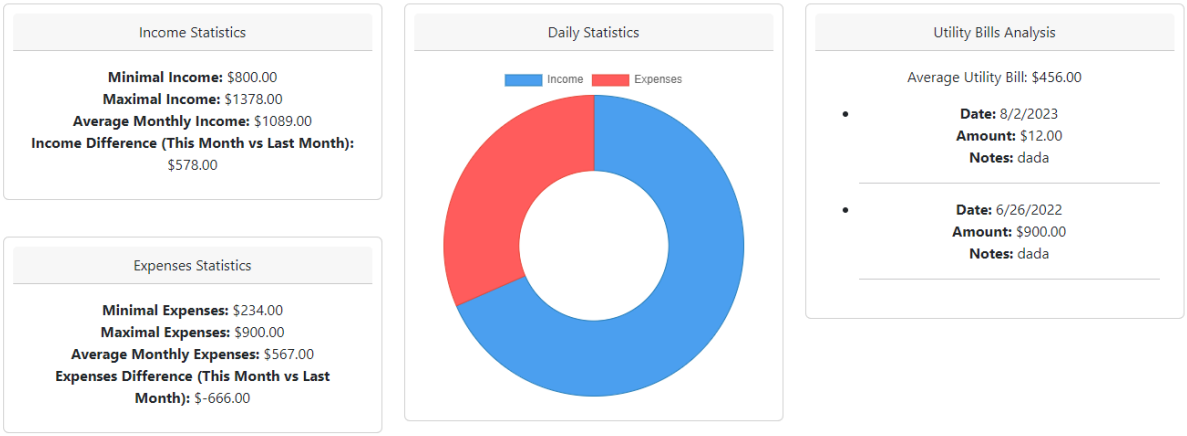
4.5 Statistika (Statistics)

Ova stranica nudi korisnicima sveobuhvatni uvid u njihovo financijsko stanje kroz široku paletu grafova, te vizualnih alata. Svaki korisnik može jednostavno pristupiti svojim dnevnim financijskim pokazateljima, kombinirajući prihode i rashode na jednom mjestu. U današnjem digitalnom dobu, matematika se koristi ne samo za složene operacije, već i za obradu i analizu podataka. U domeni financija, koristi se za filtriranje i pripremu podataka kako bi korisnici bolje pratili svoje financije. Kroz JavaScript, relevantni financijski podaci filtriraju se prema kategorijama poput prihoda i rashoda. Također, omogućena je analiza kroz različite vremenske okvire, poput tjednih, mjesečnih i godišnjih prikaza. Za dublje razumijevanje, mjesečni trendovi omogućuju praćenje promjena prihoda i rashoda tijekom vremena. Kroz jasno prikazane grafičke elemente, korisnik može uočiti ulazne i silazne linije svog financijskog stanja. S trendom kretanja troškova, korisnik može precizno filtrirati želi li analizirati samo prihode, rashode ili kombinaciju oba, uz dodatnu mogućnost odabira vremenskog perioda - bilo da je to godišnje, kvartalno, mjesečno, tjedno ili čak prikaz svega zajedno. Na kraju, svi obrađeni podaci grupiraju se za prikaz na

linijskom grafikonu, omogućujući korisnicima da jasno vide trendove i donose informirane odluke.

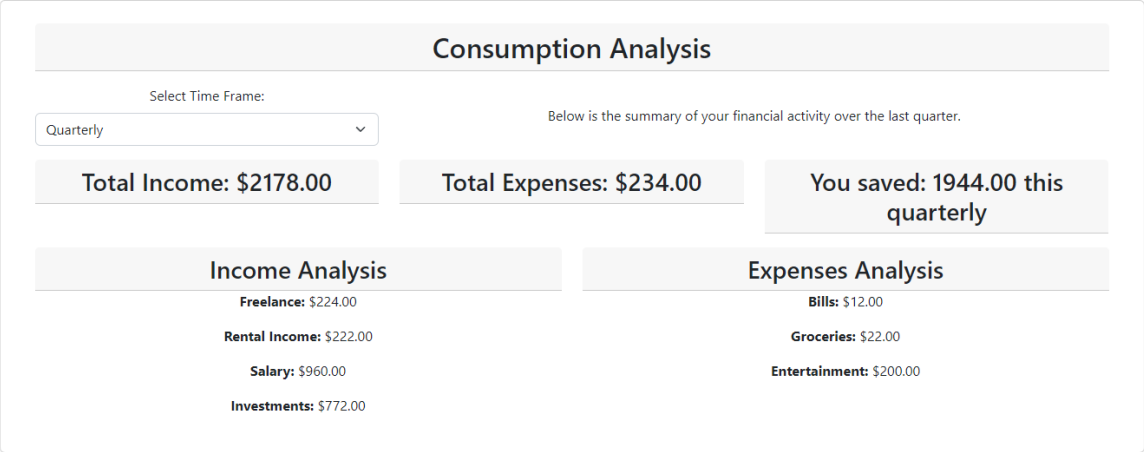
Slika 8. Graf linearnog trenda prihoda i rashoda

Da bi dobio širi kontekst, korisniku je omogućena komparativna analiza trenutnih prihoda i rashoda u odnosu na prethodni mjesec. S dnevnom statistikom, moguće je detaljno pratiti svakodnevne financijske transakcije te paralelno provjeriti status računa. Ovaj prikaz pruža korisnicima detaljan uvid u njihove financijske informacije putem raznih vizualnih alata. Dnevne statistike koje prikazuju korisnicima dnevne financijske podatke, kombinirajući prihode i rashode. Mjesečni trend prihoda i rashoda omogućuje uvid u stanje financija prošlog mjeseca u odnosu na trenutni, te prikaz minimalnih, maksimalnih i prosječnih vrijednosti prihoda i rashoda.



Slika 9. Prikaz sekundarnih praćenja vrijednosti

Na kraju, segment analize potrošnje pruža korisniku detaljan prikaz njegovog financijskog ponašanja. Nakon odabira željenog vremenskog okvira, korisnik dobiva sveobuhvatan uvid u ukupne prihode i rashode, moguću uštedu te detaljnu razgradnju troškova po kategorijama.



Slika 10. Analiza potrošnje

4.5.1 Matematika vezana za ekran statistika

Statističke funkcionalnosti aplikacije, ključne su za upravljanje niza funkcionalnosti. Jedne od osnovnih su izračuni minimalne i maksimalne vrijednosti. Minimalna vrijednost odnosi se na najmanju vrijednost u skupu podataka, dok maksimalna predstavlja najveću. Ove dvije vrijednosti pružaju uvid u raspon podataka s kojima radimo. Aplikacija koristi aritmetičku sredinu, koja predstavlja prosječnu vrijednost skupa podataka. Izračunava se zbrojem svih pojedinačnih vrijednosti i dijeljenjem njihovog ukupnog broja. Ova mjera središnje tendencije omogućava nam da brzo procijenimo opći trend ili osjećaj skupa podataka bez potrebe za analizom svake pojedinačne vrijednosti. Također, aplikacija omogućava korisnicima da promatraju linearni graf trenda kretanja (slika 8). Linearni trend grafički prikazuje kako se prihodi i rashodi mijenja tijekom vremena, pružajući vizualni uvid u kretanje i eventualne uzorke. Nažalost, trenutačno aplikacija ne pruža sveobuhvatne statističke metode deskriptivne statistike, kao što su mod, medijan, kvantil, kvartil i slično. Implementacija dodatnih statističkih metoda mogla bi dodatno poboljšati korisničko iskustvo i pružiti dublju analizu u osobne financije [6].

4.5.1.1 Osnovna analiza

-Minimalna i maksimalna vrijednost: Kroz funkciju „Math.min“ i „Math.max“ možemo utvrditi najmanje i najveće financijske vrijednosti. Na taj način korisnici mogu odmah vidjeti svoj najniži i najveći iznos troškova ili prihoda tijekom odabranog razdoblja.

Isječak programskog koda 7: Prijmer implementacije Math.min/Math.max

```
1. let transakcije = [100, 500, 300, 700];
2. let minVrijednost = Math.min(...transakcije);
3. let maxVrijednost = Math.max(...transakcije);
4.
```

-Prosječna mjesečna vrijednost: Korištenjem „reduce“ funkcije izračunava se ukupna vrijednost prihoda ili rashoda i zatim se dijeli s ukupnim brojem unosa kako bi se dobila prosječna mjesečna vrijednost. Funkcija „reduce“ prolazi kroz svaku stavku u nizu i koristi kako bi stvori jednu kumulativnu vrijednost. U našem primjeru zbraja sve stavke niza kako bi se izračunala ukupna suma.

Isječak programskog koda 8: Primjer implementacije reduce metode

```
1. let prosjek = transakcije.reduce((acc, curr) => acc + curr, 0) / transakcije.length;
2.
```

-Mjesečna razlika: Funkcija omogućuje korisnicima da vide razliku između ovog mjeseca i prethodnog mjeseca. To pruža uvid u to kako se njihovo financijsko stanje mijenja tijekom vremena.

Isječak programskog koda 9: Primjer izračuna mjesečne razlike

```
1. let razlika = transakcije[transakcije.length - 1] - transakcije[transakcije.length - 2];
```

4.5.1.2 Vizualna analiza

Dijagram prstena (Doughnut chart): Kroz „doughnut“ komponentu iz paketa react-chartjs-2, korisnici mogu vizualno usporediti svoj ukupni prihod i rashod.

Isječak programskog koda 10: Primjer podataka za 'react-chartjs-2'

```
1. const data = {
2.     labels: ['Prihod', 'Rashod'],
3.          datasets: [{
4.         data: [sumPrihod, sumRashod],
5.         backgroundColor: ['#4CAF50', '#FF5733'],
6.     }]
7. };
8.
```

4.5.1.3 Specifična analiza – računi za komunalije

Prosječni račun: Izračunava se prosječna vrijednost svih računa za komunalne usluge. Pruža korisnicima informacije o tome koliko prosječno mjesečno troše na te usluge.

Isječak programskog koda 11: Primjer izračuna prosječne vrijednosti

```
1. let komunalije = [150, 170, 160];
2. let prosjekKomunalija = komunalije.reduce((acc, curr) => acc + curr, 0) /
komunalije.length;
3.
```

4.5.1.4 Analiza potrošnje

Analiza prihoda i rashoda: Koristeći „reduce“ funkciju, moguće je kategorizirati i sumirati prihode i rashode prema određenim kategorijama.

Isječak programskog koda 12: Primjer prikaza

```
1. let kategorije = {
2.     "Hrana": [50, 60, 70],
3.     "Transport": [100, 90]
4. };
5. for (let kat in kategorije) {
6.     let suma = kategorije[kat].reduce((acc, curr) => acc + curr, 0);
7.     kategorije[kat] = suma;
8. }
```

Filter podataka po vremenskom okviru: Korisnici mogu odabrati željeni vremenski okvir (mjesečno, kvartalno, godišnje) kako bi filtrirali svoje podatke. Ova funkcija koristi logiku datuma kako bi usporedila datume unosa s odabranim vremenskim okvirom.

Isječak programskog koda 13: Primjer izračuna vremenskog okvira

```
1. // Uz pretpostavku da svaka transakcija ima datum
2. let transakcijePoMjesecima = transakcije.filter(trans => new Date(trans.datum).getMonth()
=== new Date().getMonth());
3.
```

Ukupna analiza: Na kraju, korisnik može vidjeti ukupni prihod, ukupne rashode te eventualnu uštedu ili deficit za odabrano razdoblje.

Isječak programskog koda 14: Primjer računanja ukupne analize

```
1. let ukupniPrihod = prihodi.reduce((acc, curr) => acc + curr, 0);
2. let ukupniRashod = rashodi.reduce((acc, curr) => acc + curr, 0);
3. let saldo = ukupniPrihod - ukupniRashod;
```

Uz ove matematičke funkcije i analitičke metode, korisnici mogu imati detaljan uvid u svoje financijsko stanje, razumjeti svoje financijske navike i donositi informirane odluke o upravljanju svojim financijama

4.6 Izvještaji (Reports)

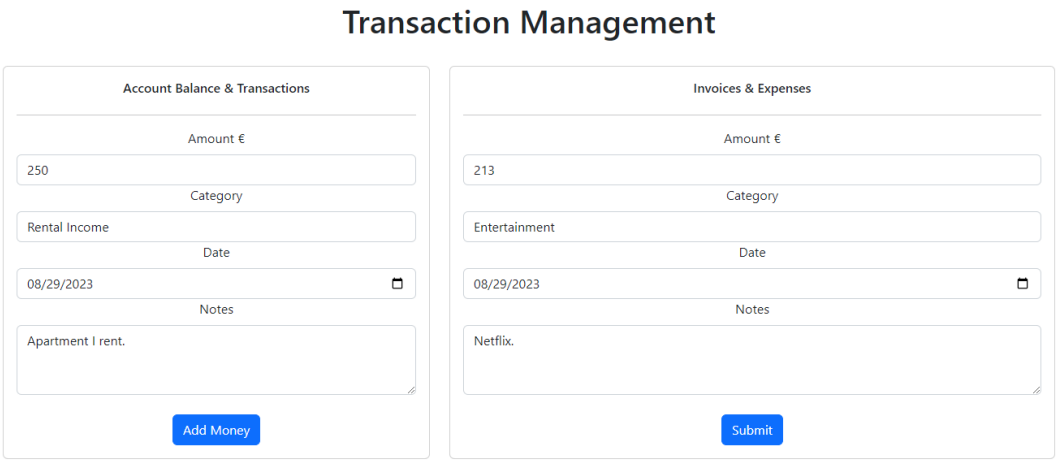
Izvještaj korisnicima omogućuje pregled svih računa za prihode i rashode. Korisnici mogu urediti ili obrisati pojedini račun. Prihodi i rashodi su jasno odvojeni kako bi se izbjegla potencijalna konfuzija. Dodatno, sa strane se nalazi tražilica koja omogućava filtriranje prema različitim parametrima, uključujući naziv, kategoriju i datumski raspon. Također je dostupan gumb za izvoz podataka u excel tablicu.

Slika 11. Prikaz report tablice

4.7 Upravitelj transakcijama (Transaction Manager)

Ova stranica omogućava korisniku jednostavan unos svojih financijskih transakcija, bilo da se radi o prihodima ili rashodima. Korisnik specificira iznos, odabire relevantnu kategoriju, bilježi datum te dodaje kratku opisnu napomenu za svaku transakciju.

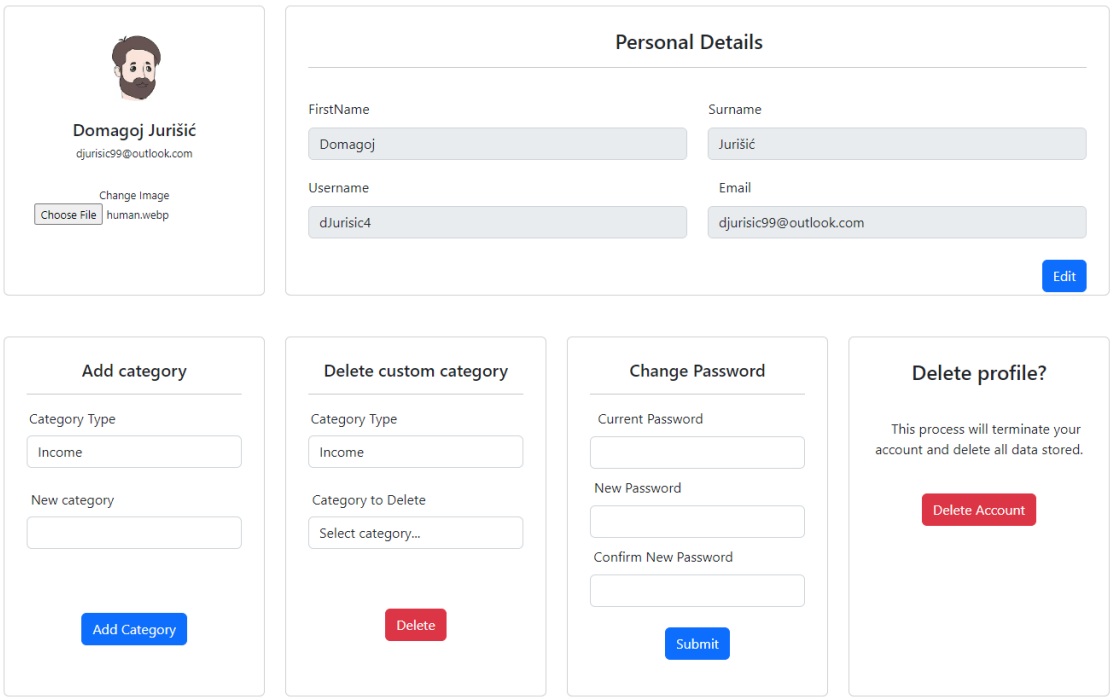
Za veću prilagodljivost, unaprijed definirane kategorije su dostupne za oba tipa transakcija – prihode i rashode. No, ne završava se tu. Prepoznajući raznolikost i specifičnost svakodnevnih financijskih transakcija, platforma omogućava korisnicima da kreiraju i svoje vlastite kategorije. Jednom kada se takva prilagođena kategorija stvori, automatski postaje dostupna u padajućem izborniku, olakšavajući buduće unose. Ovaj digitalni alat je dizajniran s misijom da pojednostavi i organizira financijske transakcije korisnika, pružajući im veći nadzor i uvid u njihovo financijsko stanje. Kroz intuitivno sučelje, korisnici mogu lako pratiti i analizirati svoje prihode i rashode, te donositi informirane odluke vezane za svoje financije.



Slika 12. Transakcijski ekran

4.8 Postavke (Settings)

Postavke korisnicima otvaraju vrata sofisticiranom sučelju namijenjenom preciznom prilagođavanja osobnog profila i financijskih kategorija. Kombinacija estetike i funkcionalnosti teži pružanju vrhunskih iskustava personalizacije uz maksimalnu sigurnost. Korisnici ne samo da mogu osvježiti svoju profilnu fotografiju i ažurirati osobne podatke, te mijenjati lozinku ili čak trajno ukloniti svoj profil. Nadalje, pružena je sloboda kreiranja nove kategorije ili eliminacije postojeće, ukoliko procijene da im više nije od koristi.



Slika 13. Prikaz ekrana Postavke

5. Usporedba „Personal Finance Manager“ aplikacije sa „You Need A Budget“ aplikacijom

Postoji niz aplikacija koje korisnicima pružaju različite alate i funkcionalnosti kada pričamo o vođenju osobnih financija. Jedna od njih je i "You Need A Budget", često skraćeno kao YNAB. Ova sofisticirana aplikacija za upravljanje financijama nudi korisnicima alate za praćenje rashoda, postavljanje financijskih ciljeva i uspostavljanje budžeta. Fokusirajući se na filozofiju da svaki dolar ima svoju svrhu, YNAB potiče korisnike na svjesnost svog financijskog stanja i aktivno upravljaju svojim sredstvima. Uz intuitivno sučelje i mogućnost sinkronizacije s bankovnim računima, aplikacija olakšava korisnicima praćenje i optimizaciju njihovih financija [7].

Ali kako se „Personal Finance Manager“ mjeri u odnosu na YNAB? Kako bismo bolje razumjeli razlike i sličnosti između ovih dviju aplikacija, prikazat ćemo detaljnu usporedbu u tablici koja slijedi.

Tablica 2. Usporedba „Personal Finance Manager“ aplikacije sa „YNAD“ aplikacijom.

	Personal Finance Manager	You Need A Budget
Pristup preko PC-a, Mobitela. Tableta	NE	DA
Izvještaj o troškovima i neto vrijednosti	DA	DA
Praćenje ciljeva	NE	DA
Praćenje Statistike	DA	DA
Izvoz podataka	DA	DA
Uređivanje prilagođenih kategorija	DA	DA
Besplatno	DA	NE

Tablica iznad pažljivo analizira i uspoređuje funkcionalnosti dviju popularnih aplikacija za upravljanje osobnim financijama: „Personal Finance Manager“ i „You Need

A Budget“. Dok „Personal Finance Manager“ nudi širok spektar alata, uključujući izvoz u Excel formatu i mogućnost prilagođavanja kategorija bez dodatnih troškova, „You Need A Budget“ se izdvaja svojom sveobuhvatnom platformom koja omogućuje pristup preko različitih uređaja i naprednim praćenjem ciljeva. Usprkos sličnostima, svaka aplikacija ima svoje jedinstvene značajke koje će korisnicima pomoći da donesu informiranu odluku o tome koja aplikacija najbolje odgovara njihovim potrebama.

6.Zaključak

U eri globalizacije i ubrzanog tehnološkog napretka, sposobnost efikasnog upravljanja financijama postaje ključna za ostvarivanje osobnih i profesionalnih ciljeva. I dok financijske strategije evoluiraju od dana drevnih civilizacija, bitno je razumjeti da osnovna načela, poput stabilnosti i održivosti, ostaju nepromijenjena. Aplikacija „Personal Finance Manager“ nije samo odraz tog evolutivnog puta, već i most između prošlih znanja i suvremene tehnologije. Ona uspješno kombinira tradicionalne financijske metode s inovativnim rješenjima, pružajući korisnicima platformu koja je intuitivna, transparentna i prilagodljiva. S obzirom na to, ovakve aplikacije ne samo da pojednostavljuju financijsko upravljanje, već i potiču kulturu financijske pismenosti. Ova transformacija omogućava svakom pojedincu, neovisno o njegovom predznanju ili iskustvu, da donosi informirane odluke, optimizira svoje resurse i prilagodi se stalno mijenjajućem financijskom okruženju. Kako tehnologija nastavlja napredovati, važno je da nastavimo s razvojem alata „Personal Finance Manager“ kako bi pružili korisnicima lakše snalaženje u svijetu financija. S obzirom na sve veću uporabu mobilnih uređaja i potrebu za stalnim praćenjem financija u pokretu, implemetacija „Personal Finance Manager“ aplikacije na mobitele i tablete postala je neizbježna nadogradnja za budućnost. Također, uvođenje funkcionalnosti praćenja omogućit će korisnicima da postave i prate svoje financijske ciljeve, dodatno potičući odgovorno i proaktivno upravljanje sredstvima.

7. Popis literature

[1] Richard Brown: A History of Accounting and Accountants

[2] Facebook Inc. (2021). : React – A JavaScript library for building user interfaces.

[3] Jovanovic, P., & Goodwill, J. (2020). : Pro Spring Boot 2: An Authoritative Guide to Building Microservices, Web and Enterprise Applications, and Best Practices. Apress.

[4] Bell, S. (2019). : MySQL for Beginners. Independent Publishing.

[5] Oracle Corporation. (2021). : MySQL Workbench – Database Design & Modeling. Dostupno na: https://www.mysql.com/products/workbench/

[6] Jasna Horvat i Josipa Mijoč (2014): Osnove Statistike (drugo izdanje)

[7] https://www.ynab.com/features/

8. Popis tablica

9. Popis slika

Veleučilište u Virovitici

OBRAZAC 5

IZJAVA O AUTORSTVU

Ja, Domagoj Jurišić

izjavljujem da sam autor/ica završnog/diplomskog rada pod nazivom

Upravljanje financijama s personal finance manager web aplikacijom

Svojim vlastoručnim potpisom jamčim sljedeće:

- da je predani završni/diplomski rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu, a što pokazuju korištene bilješke i bibliografija,
- da su radovi i mišljenja drugih autora/ica, koje sam u svom radu koristio/la, jasno navedeni i označeni u tekstu te u popisu literature,
- da sam u radu poštivao/la pravila znanstvenog i akademskog rada.

Potpis studenta/ice

Jurišić

Veleučilište u Virovitici

OBRAZAC 6

ODOBRENJE ZA OBJAVLJIVANJE ZAVRŠNOG/DIPLOMSKOG RADA U DIGITALNOM REPOZITORIJU

Ja Domagoj Jurišić

dajem odobrenje za objavljivanje mog autorskog završnog/diplomskog rada u javno dostupnom digitalnom repozitoriju Veleučilišta u Virovitici sadržanom u Dabar (Digitalni akademski arhivi i repozitoriji) te u javnoj internetskoj bazi završnih radova Nacionalne i sveučilišne knjižnice bez vremenskog ograničenja i novčane nadoknade, a u skladu s odredbama članka 58. stavka 5., odnosno članka 59. stavka 4. Zakona o visokom obrazovanju i znanstvenoj djelatnosti (NN 119/22).
Potvrđujem da je za pohranu dostavljena završna verzija obranjenog i dovršenog završnog/diplomskog rada. Ovom izjavom, kao autor navedenog rada dajem odobrenje i da se moj rad, bez naknade, trajno javno objavi i besplatno učini dostupnim na sljedeći način:

a) Rad u otvorenom pristupu
b) Rad dostupan nakon: _____ (upisati datum nakon kojeg želite da rad bude dostupan)
c) Pristup svim korisnicima iz sustava znanosti i visokog obrazovanja RH
d) Pristup korisnicima matične ustanove
e) Rad nije dostupan (*u slučaju potrebe dodatnog ograničavanja pristupa Vašem završnom/diplomskom radu, podnosi se pisani obrazloženi zahtjev*).

Potpis studenta/ice

Jurišić

U Virovitici, 06.09.2023.