**SVEUČILŠTE U MOSTARU**

**FAKULTET STROJARSTVA I RAČUNARSTVA**

PROJEKTNI RAD PREDMETA

PROJEKTIRANJE INFORMACIJSKIH SUSTAVA

**Upravljanje troškovima**

**TIM 4:**

Marin Bešlo

Mirko Bagarić

Petar Vukoja

Mateo Gagro

SADRŽAJ

[1 UVOD 1](#__RefHeading___Toc276_623309928)

[2 Analiza 2](#__RefHeading___Toc278_623309928)

[2.1Što jenaša aplikacija? 2](#__RefHeading___Toc280_623309928)

[2.2Trenutno stanje 2](#__RefHeading___Toc282_623309928)

[2.3Ciljevi 2](#__RefHeading___Toc284_623309928)

[2.4Workflow – primjer 2](#__RefHeading___Toc286_623309928)

[3 Specifikacija zahtjeva 3](#__RefHeading___Toc288_623309928)

[3.1Funkcionalni zahtjevi 3](#__RefHeading___Toc290_623309928)

[3.2Nefunkcionalni zahtjevi 4](#__RefHeading___Toc292_623309928)

[4 Use case templates 5](#__RefHeading___Toc294_623309928)

[4.1Use case sustava 5](#__RefHeading___Toc296_623309928)

[5 UML class diagram 6](#__RefHeading___Toc300_623309928)

[6 Baza podataka SQLite 8](#__RefHeading___Toc732_623309928)

**Upravljanje troškovima**

# UVOD

Svatko ima određeni odnos sa novcem. Potreban nam je da bi kupili stvari koje želimo, ali zadržati ga je najveći problem. Moramo platiti račune, platiti hranu koju jedemo, i ako nam nešto ostane to potrošimo na zabavu. Upravljanje novcem vam ne ide baš najbolje i imate problema sa njim i željeli biste dovesti svoje financije u red ? Riješenje za vas je naša aplikacija „Upravljanje troškovima“.

Aplikacija koja će vam pomoći sačuvati novac, budžet za buduće troškove, a također izbjeći naknade koje proizlaze iz prekoračenja ili naknada za zakasninu. Pronaći dobar budžet nije lak posao, ali sa našom aplikacijom svoje financije ćete zadržati pod kontrolom. Možete koristiti našu aplikaciju da pretvorite vaš mobitel u privatni financijski savjetnik koji vam može pomoći sa novcem, da znate kada ga trošiti, kada ga beskorisno trošite i kako da povećate budžet.

Nema potrebe da trošite sat vremena svaki dan radeći na budžetu kada naša aplikacija automatski to radi za vas i pomaže vam da vidite svoje navike u kojima beskorisno trošite novac i upozori vas na greške pri trošenju prije nego što se one dogode. Naša aplikacija vam automatski sačuvava novac i pomaže vam da napravite budžet za crne dane bez promjena na glavnom budžetu.

# Analiza

## Što jenaša aplikacija?

Naša aplikacija je mobilna android aplikacija za upravljanje troškovima koja vam može pomoći sačuvati novac, izbjeći plaćanje nepotrebnih naknada i postaviti vas na put na kojem ćete izgraditi budžet za hitne slučajeve i pokazati vam ostale pametne poteze s novcem.

Možete koristiti našu aplikaciju da pretvorite vaš mobitel u privatni financijski savjetnik koji vam može pomoći sa novcem, da znate kada ga trošiti, kada ga beskorisno trošite i kako da povećate budžet.Pronaći dobar budžet nije lak posao, ali sa našom aplikacijom svoje financije ćete zadržati pod kontrolom.

## Trenutno stanje

Aplikacija je trenutno funkcionalna i omogućuje korisniku osnovne radnje poput unosa transakcija, upravljanja računima, te pregledom transakcija na dnevnoj i mjesečnoj razini, te ukupan pregled od početka korištenja aplikacije.

## Ciljevi

Cilj je prilagoditi aplikaciju maksimalno korisnicima, rješavati potencijalne bugove, te dodati dodatne filtere pomoću kojih bi mogli još detaljnije pratiti transakcije koje korisnik radi. Nakon prve verzije očekujemo povratnu informaciju korisnika te se nadamo uvažiti prijedloge istih.

## Workflow – primjer

Navodimo primjer korištenja aplikacije:

1. Korisnik naglašava određenu svotu novca i određuje jeli ta vrijednost prihod ili rashod.
2. Bira kategoriju na kojoj je potrošio ili dobio određenu svotu novca(npr. Hrana, Zabava, Poklon, Prijevoz itd.)
3. Odabire račun na kojem želi da se ta vrijednost obračuna i po želji dodaje bilješku vezanu za tu vrijednost.
4. Kada spremimo prihod/rashod on se obračuna u određeni račun u kojem možemo vidjeti trenutni budžet i detaljno pregledavati troškove.

# Specifikacija zahtjeva

Čitav projekt je kreiran po određenim zahtjevima. Glavni uvjet za korištenje aplikacije je android platforma. Sama aplikacija, ali i popratna dokumentacija i ostali dokumenti imaju neke stavke koje je trebalo ispuniti. Oni su sljedeći:

Unutar aplikacije treba implementirati:

* Upravljanje računima (gotovina, kartice i sl.)
* Kreiranje, ažuriranje i brisanje kategorija potrošnje
* Kreiranje, ažuriranje i brisanje pojedinog troška
* Podrška više valuta
* Reporting funkcija - pregled i pretraživanje troškova uz korištenje filtera (po kategoriji, po cijeni, nazivu i raspoloživosti, anima, tjednima,mjesecima i sl itd)
* itd…

*Kreirati*:

* Projektnu dokumentaciju koja sadrži:
  + Specifikaciju zahtjeva putem UML modela slučajeva korištenja
  + UML modele klasa i prijedlog-specifikaciju rješenja
  + Smjernice za daljnji razvoj aplikacije .
  + GITHUB repozitorij projekta
  + Korištenje alata za kolaboraciju i praćenje projekata

## Funkcionalni zahtjevi

Funkcionalni zahtjevi predstavljaju izjave o uslugama koje sustav mora pružiti, te kako bi se sustav morao ponašati u određenim situacijama. Funkcionalni zahtjevi aplikacije za upravljanje troskovima:

1. Aplikacija mora imati glavni menu sa dnevnim pregledom i posljednjim izmjenama.
2. Aplikacija mora imati mogućnost ulaska u polje „posljednje izmjene“ gdje možemo pobliže vidjeti sve izmjene.
3. Aplikacija mora dati mogućnost mjenjanja trenutnog računa kojeg želimo vidjeti u glavnom meniju.
4. Aplikacija mora imati formu za dodavanje novog prihoda/rashoda.
5. Aplikacija mora imati polje za dodavanje određenog iznosa transakcije.
6. Aplikacija mora imati dugme za izbor jeli transakcija prihod ili rashod.
7. Apikacija mora dati mogućnost izbora kategorije u koju dodajemo novu transakciju.
8. Apikacija mora dati mogućnost izbora računa u kojem želimo obračunat novu transakciju.
9. Aplikacija mora imati mogućnosti izbora dana(datum) na kojeg smo obavili određenu transakciju.
10. Aplikacija mora imati mogućnost dodavanja bilješke za novu transakciju.
11. Aplikacija mora dati mogućnost spremanja i odbacivanja transakcije na kojoj trenutno radimo.
12. Aplikacija mora omogućiti pregled mjesečnih i ukupnih troškova.
13. Aplikacija mora omogućiti pregled kategorija u kojima smo obavljali transakcije.
14. Aplikacija mora imati mogućnost da odaberemo određenu valutu koju želimo koristiti.
15. Aplikacija mora imati mogućnost pregleda informacija o proizvodu.

## Nefunkcionalni zahtjevi

Nefunkcionalni zahtjevi predstavljaju ograničenja u uslugama i funkcijama. Nefunkcionalni zahtjevi aplikacije za upravljanje troškovima:

1. Sučelje mora biti jednostavno
2. Sve nepravilnosti u radu aplikacije treba otkriti na vrijeme
3. Neovlašteni pristup aplikaciji treba biti onemogućen
4. Baza podataka
5. Sama aplikacija zauzima manje od 5 MB prostora
6. Aplikacija treba biti dostupna 24 sata na dan

# Use case templates

## Use case sustava

U ovom dijelu smo dali detaljniji pregled načina rada aplikacije. Opisani su uvjeti u kojima se događa određena radnja pri korištenju aplikacije. Tekst s detaljnim opisom se nalazi u mapi "UML" koja je priložena uz ovaj projekt. Tu možete pronaći opis use case, tko su korisnici, detaljnije opise pojedinih akcija, uvjete odvijanja akcija, tijek akcije, te dodatne stvari potrebne za korištenje same aplikacije.

# UML class diagram



Osnovni zadatak aplikacije je korisniku omogućiti upravljanje troškovima i prihodima. Pri unosu transakcije, dužan je unijeti iznos same transakcije, označiti radi li se o prihodu, ili pak rashodu, odabrati kategoriju na koju se transakcija odnosi, datum, račun koji koristi, te bilješku gdje može opisati malo pobliže o samoj transakciji. Ukoliko nije zadovoljan unosom, u mogućnosti je odbaciti upisane vrijednosti, ili nakon spremanja ponovno ispraviti. Ispravljanje unosa (update, engl.) se vrši dužim pritiskom na sam unos u listi rubrike "Posljednje izmjene/Pregled transakcija". Korisnik ima mogućnost i u potpunosti ukloniti pohranjene unose dugim pritiskom na unos, te odabirom opcije "Izbriši".

Korisnik je u mogućnosti u potpunosti upravljati svojim troškovima. Može kreirati nove račune u rubrici "Pregled računa" (Accounts overview, engl.), te dodavati nove kategorije. Kategorije dodaje klikom na gornju desnu ikonu s početnog ekrana, te odabira opciju kategorije, gdje ima mogućnost unosa novih, ili pak brisanja starih kategorija. Opcija "Valute" (Currency, engl.) omogućuje odabir valute koju želi koristiti.

Klikom na lijevi gornji gumb (Navigation drawer), korisniku se otvara mogućnost da pregleda izdatke za dnevnu, mjesečnu ili pak ukupnu razinu korištenja aplikacije.

Na početnom ekranu, korisniku su prikazane današnje transakcije, pregled računa, te posljednjih 5 unosa. Klikom na račune otvaraju mu se gore navedene mogućnosti upravljanja istih. Ukoliko pak želi pregled svih unosa, potrebno je odabrati rubriku "Posljednje izmjene". Dodavanje novog unosa se obavlja klikom na donji desni gumb (Floating button, engl.).

# Baza podataka SQLite

**Opis tablica u relacijskom modelu**

Popis tablica:

- Account (mogućnost upravljanja računima)

- Charge (mogućnost upravljanja unosima troškova/prihoda)

- Category (mogućnost upravljanja kategorijama)

- Currency (mogućnost upravljanja valutama)

|  |
| --- |
| public static final String DATABASE\_NAME = "Troskovi.db"; |
| //TABLE\_1 - Category |
| public static final String TABLE\_1 = "category"; |
| public static final String COL\_11 = "category\_id"; |
| public static final String COL\_12 = "category\_name"; |
| //TABLE\_2 - Account |
| public static final String TABLE\_2 = "account"; |
| public static final String COL\_21 = "account\_id"; |
| public static final String COL\_22 = "account\_name"; |
| public static final String COL\_23 = "account\_value"; |
| //TABLE\_3 - Charge |
| public static final String TABLE\_3 = "charge"; |
| public static final String COL\_31 = "charge\_id"; |
| public static final String COL\_32 = "charge\_price"; |
| public static final String COL\_33 = "charge\_type"; |
| public static final String COL\_34 = "charge\_date"; |
| public static final String COL\_35 = "charge\_note"; |
| public static final String COL\_36 = "charge\_account\_id"; |
| public static final String COL\_37 = "charge\_category\_id"; |
| //TABLE\_4 - Currency |
| public static final String TABLE\_4 = "currency"; |
| public static final String COL\_41 = "currency\_id"; |
| public static final String COL\_42 = "currency\_name"; |

Detaljan pregled tablica u bazi podataka možete pogledati u samom kodu aplikacije, u klasi DBHelper.java gdje su prikazane kolone tablica te popis svih upita (query, engl.) koji su korišteni pri izradi same aplikacije.