SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

**Davorin Horvat**

**Dominik Lacković**

**Blaža Marinić**

**Filip Strahija**

APLIKACIJA PERSONAL FINANCE

tehnička dokumentacija IZ KOLEGIJA ANALIZA I RAZVOJ PROGRAMA

Varaždin, 2017.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Oznaka tima: AIR1605

Davorin Horvat, 44494/15-R

Dominik Lacković, 45290/16-R

Blaža Marinić, 45307/16-R

Filip Strahija, 45330/16-R

GitHub repozitorij: <https://github.com/dhorvat3/Personal-Finance>

APLIKACIJA PERSONAL FINANCE

tehnička dokumentacija IZ KOLEGIJA ANALIZA I RAZVOJ PROGRAMA

Mentor:

Doc. dr. sc. Zlatko Stapić

Varaždin, veljača 2017.

**Sadržaj**

[1. Uvod 1](#_Toc474909417)

[1.1. Svrha 1](#_Toc474909418)

[1.2. Opseg 1](#_Toc474909419)

[1.3. Definicije, akronimi, kratice 1](#_Toc474909420)

[1.4. Reference 1](#_Toc474909421)

[1.5. Pregled 1](#_Toc474909422)

[2. Opis aplikacije 2](#_Toc474909423)

[2.1. Perspektiva proizvoda 2](#_Toc474909424)

[2.2. Funkcionalnosti aplikacije 2](#_Toc474909425)

[2.3. Korisnički zahtjevi 3](#_Toc474909426)

[2.4. Ograničenja 3](#_Toc474909427)

[3. Specifični zahtjevi 4](#_Toc474909428)

[3.1. Prikaz arhitekture aplikacije 4](#_Toc474909429)

[3.2. ERA model 6](#_Toc474909430)

[3.3. Opis funkcionalnosti aplikacije 7](#_Toc474909431)

[3.3.1. Detaljan opis slučajeva korištenja 7](#_Toc474909432)

[3.3.1.1. Slučaj korištenja Registracija korisnika 7](#_Toc474909433)

[3.3.1.2. Slučaj korištenja Prijava korisnika 8](#_Toc474909434)

[3.3.1.3. Slučaj korištenja Profil korisnika 9](#_Toc474909435)

[3.3.1.4. Slučaj korištenja Dnevni prihodi i rashodi 10](#_Toc474909436)

[3.3.1.5. Slučaj korištenja Kategorije 11](#_Toc474909437)

[3.3.1.6. Slučaj korištenja Dnevnik 12](#_Toc474909438)

[3.3.1.7. Slučaj korištenja Statistika 13](#_Toc474909439)

[3.3.1.8. Slučaj korištenja Lista obveza 14](#_Toc474909440)

[3.3.1.9. Slučaj korištenja Odjava korisnika 15](#_Toc474909441)

[3.3.1.10. Slučaj korištenja Sinkronizacija 15](#_Toc474909442)

[4. Mockup 16](#_Toc474909444)

[5. Dijagram klasa 21](#_Toc474909445)

[6. Struktura programskog koda 22](#_Toc474909446)

[7. Specifikacija web servisa 23](#_Toc474909447)

[8. Pristup u radu s podacima 36](#_Toc474909448)

[9. Testiranje 37](#_Toc474909449)

[9.1. Uputstvo za testiranje aplikacije 37](#_Toc474909450)

[9.2. UI testovi 37](#_Toc474909451)

1. Uvod

Dokumentacija sadrži specifikaciju softverskih zahtjeva prema IEEE Std 830-1998 standardu. Poglavlje sadrži pregled zahtjeva softvera, sadržanih u sljedećim podpoglavljima: svrha, opseg, definicije, reference i pregled.

* 1. Svrha

Svrha dokumenta je opis funkcionalnosti aplikacije i uloga podržanih od same aplikacije. Specifikacija zahtjeva namijenjena je članovima razvojnog tima i mentorima za lakše razumijevanje programskog proizvoda te analizu implementacije same aplikacije

* 1. Opseg

Personal Finance je aplikacija za vođenje osobnih financija za mobilne uređaje temeljene na Android operacijskom sustavu. Personal Finance je mobilna aplikacija namijenjena korisnicima koji žele pratiti kretanje vlastitih financija u smislu svakodnevnog unosa prihoda i rashoda. Cilj aplikacije je olakšati korisniku uvid na koje vrste rashoda mu odlazi najveći dio financija, odnosno od koje vrste prihoda dolazi najveći dio financija. Podržava ulogu korisnika.

* 1. Definicije, akronimi, kratice

Korisnik: korisnik aplikacije.

* 1. Reference

IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

* 1. Pregled

Sljedeće poglavlje sadrži opis aplikacije, prikazan kroz sljedeća podpoglavlja: perspektiva proizvoda, funkcionalnosti aplikacije, korisnički zahtjevi, ograničenja.

1. Opis aplikacije

Opis samog proizvoda, korisničkih zahtjeva i funkcionalnosti.

* 1. Perspektiva proizvoda

Veliki broj ljudi voli pratiti kretanje vlastitih financija, koliko troše i koliko zarađuju, kako bi pratili s koliko novca raspolažu te što bi mogli poboljšati na tom području. No i dalje mnogi to rade uz pomoć olovke i papira, smatramo da bi im naša aplikacija uvelike olakšala u praćenju financija, a zbog svoje jednostavnosti bi ju mogao koristiti svatko tko raspolaže Android pametnim telefonom.

* 1. Funkcionalnosti aplikacije

U nastavku su nabrojane funkcionalnosti koje sustav treba zadovoljiti te kratak opis svake:

1. Registracija i prijava korisnika – prilikom prvog ulaska u aplikaciju korisnik se mora registrirati unoseći e-mail, korisničko ime i lozinku, dok se prilikom svakog slijedećeg ulaska u aplikaciju korisnik mora prijaviti unoseći korisničko ime i lozinku odabrane pri registraciji.
2. Kreiranje i uređivanje profila korisnika – korisnik u svoj profil unosi općenite podatke o sebi, kao što su ime, prezime, spol, datum rođenja, email itd.
3. Pregled i unos dnevnih prihoda i rashoda – nakon prijave korisniku se otvara glavni prozor na kojem se nalazi lista unesenih prihoda i rashoda za taj dan. Korisnik ima mogućnost dodavanja stavki.
4. Odabir kategorija prihoda i rashoda – korisnik može svrstati prihode i rashode u kategorije koje je sam kreirao.
5. Pregled prihoda i rashoda po datumu (dnevnik) – odabirom željenog datuma ili kategorije, korisniku se prikazuju prihodi i rashodi koje je unio toga dana, tj. za tu kategoriju.
6. Pregled statistike prihoda i rashoda – statistički prikaz prihoda i rashoda u obliku grafova.
7. Pregled i unos u listu obveza – korisnik ima mogućnost unijeti buduću obvezu u aplikaciju, koja će ga podsjetiti na istu kada za to dođe vrijeme, također može pregledati unesene obveze.
8. Odjava korisnika – korisnik može odjaviti iz aplikacije nakon korištenja.
9. Sinkronizacija – Web sinkronizacija.
   1. Korisnički zahtjevi

Za korištenje aplikacije potreban je Android mobilni uređaj s minimalnim API Level-om 19.

* 1. Ograničenja

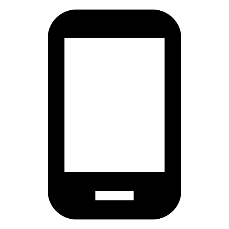
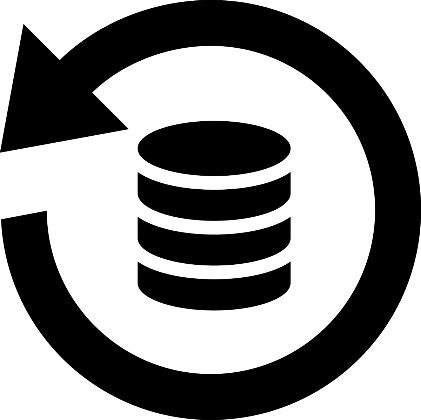
Aplikacija je vezana samo za Android uređaje s minimalnim API Level-om od 19.

1. Specifični zahtjevi

U nastavku slijede prikaz arhitekture sustava te detaljan prikaz funkcionalnosti aplikacije.

* 1. Prikaz arhitekture aplikacije

Personal Finance je mobilna aplikacija koja koristi web servis kako bi dohvatila, dodala ili promijenila podatke u bazi. Na web poslužitelj se spajaju korisnici putem mobilnog uređaja (aplikacije) te vrše pozive prema web servisu, koji se povezuje s bazom podataka. Sljedeća slika prikazuje arhitekturu sustava u kojem aplikacija radi i s kojim je u interakciji.





Korisnici Mobilni uređaj Web servis Baza podataka

Slika 1: Prikaz arhitekture sustava

Slijedi detaljniji prikaz arhitekture aplikacije. Na slici 2 prikazali smo aplikacijske slojeve kroz tri boje. Plava boja simbolizira temelje na kojima je građena aplikacija, odnosno podatkovni sloj u kojem se nalaze podaci potrebni u radi aplikacije i podaci koji se prokazuju korisniku. Zelena i bijela boja simboliziraju kombinaciju logičkog sloja u kojem se obavljaju operacije nad web servisom i bazom podataka te sloja prezentacije u kojem su ti podaci prikazani korisniku.

Spajanje na web servis

Prikaz

Unos

Prikaz

Izmjena

Izmjena

Unos

Brisanje

Izmjena

Izmjena

Unos

Brisanje

Unos

Brisanje

Dnevnik

Profil

Kategorije

Lista obveza

Statistika

Dnevni prohodi i rashodi

Personal Finance

Web servis

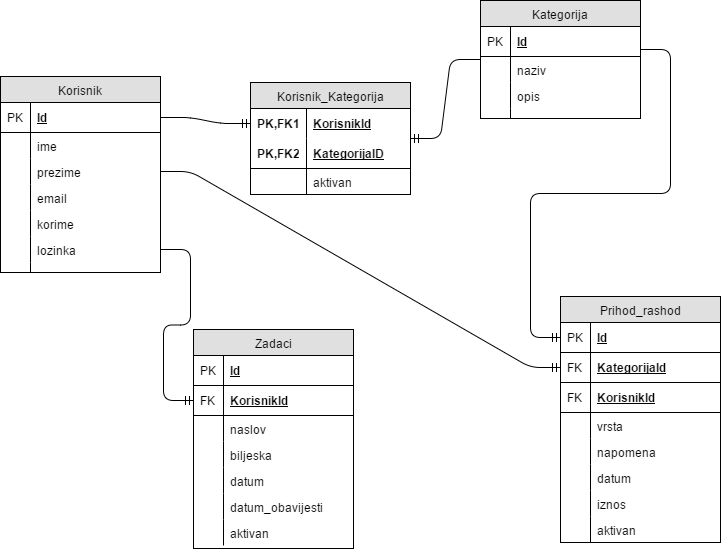
Spajanje na bazu podataka

Baza podataka

Slika 2: Detaljni prikaz arhitekture sustava

* 1. ERA model

Na slici 3 prikazan je ERA model trenutne faze aplikacije. ERA model sastoji se od četiri entiteta: Korisnik, Korisnik\_Katgorija, Kategorija, Prihod\_rashod. Kroz entitet Korisnik bilježe se registrirani korisnici i njihovi osobni podaci (ime, prezime, email, korisničko ime i lozinka), od kojih se neki prosljeđuju na profil korisnika koji se naknadno može mijenjati. Svrha aplikacije je unos i prikaz unesenih prihoda i rashoda, tome služi entitet Prihod\_rashod u kojeg se spremaju podaci o prihodu ili rashodu kojeg je korisnik unio (vrsta, napomena, datum, iznos i atribut aktivan koji je potreban ukoliko korisnik obriše neki od prihoda ili rashoda). Korisnik također može odabrati kategoriju i unijeti novu kategoriju po svom izboru kroz atribute u tablici (naziv, opis). Svaki korisnik može odabrati između više kategorija, i svaku kategoriju može odabrati više korisnika, stoga je između te dvije tablice dodan slabi entitet Korisnik\_Kategorija koji također ima atribut aktivan s istom svrhom kao ranije spomenuti, samo što se u ovom slučaju brišu kategorije.



Slika 3: ERA model

* 1. Opis funkcionalnosti aplikacije

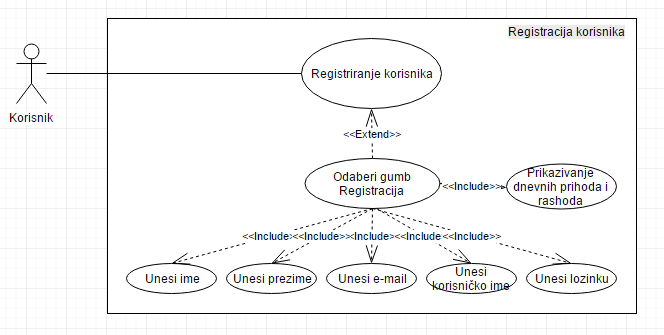
Prikaz navedenih funkcionalnosti može se vidjeti kroz detaljne opise slučajeva korištenja te prikaza dijagrama istih.

* + 1. Detaljan opis slučajeva korištenja

|  |  |
| --- | --- |
| **ID** | **Opis scenarija korištenja** |
| UC-1 | Registracija korisnika |
| UC-2 | Prijava korisnika |
| UC-3 | Profil korisnika |
| UC-4 | Dnevni prohodi i rashodi |
| UC-5 | Kategorije |
| UC-6 | Dnevnik |
| UC-7 | Statistika |
| UC-8 | Lista obveza |
| UC-9 | Odjava korisnika |
| UC-10 | Sinkronizacija |

* + - 1. Slučaj korištenja Registracija korisnika

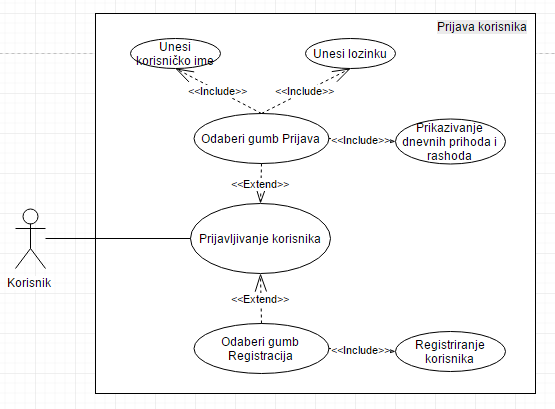
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-1 | Registracija korisnika |
| **Opis scenarija korištenja:**  Registracija korisnika u aplikaciju uz unos: imena, prezimena, e-maila, korisničkog imena i lozinke. | |
| **Preduvjeti:**   * Korisnik nema kreiran korisnički račun s istim podacima * Android uređaj prikazuje ekran Registracija aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Registracija na početnom ekranu aplikacije. |
| 2 | Prikazuje se ekran Registracija s poljima koje je potrebno popuniti. |
| 3 | Korisnik potvrđuje unesene podatke. |



Slika 4: Dijagram slučaja korištenja Registracija korisnika

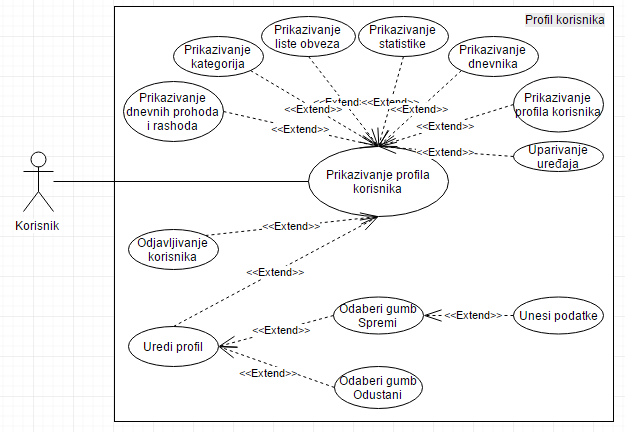
* + - 1. Slučaj korištenja Prijava korisnika

|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-2 | Prijava korisnika |
| **Opis scenarija korištenja:**  Prijava korisnika u aplikaciju uz unos: korisničkog imena i lozinke. | |
| **Preduvjeti:**   * Korisnik ima kreiran korisnički račun * Android uređaj prikazuje ekran Prijava aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Prikazuje se ekran Prijava s poljima koje je potrebno popuniti. |
| 2 | Korisnik potvrđuje unesene podatke. |

  
Slika 5: Dijagram slučaja korištenja Prijava korisnika

* + - 1. Slučaj korištenja Profil korisnika

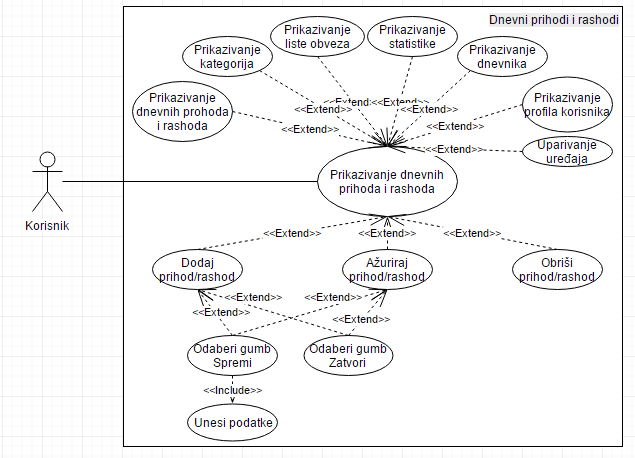
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-3 | Profil korisnika |
| **Opis scenarija korištenja:**  Pregledavanje profila od strane korisnika. Izmjena osobnih podataka. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Moj profil aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Moj profil iz navigacije. |
| 2 | Prikazuje se ekran Moj profil s podacima o korisniku. |
| 3 | Korisnik odabire opciju uređivanja profila. |
| 4 | Korisnik može izmijeniti postojeće podatke. |
| 5 | Korisnik potvrđuje unesene podatke ili odustaje od izmjene. |



Slika 6: Dijagram slučaja korištenja Profil korisnika

* + - 1. Slučaj korištenja Dnevni prihodi i rashodi

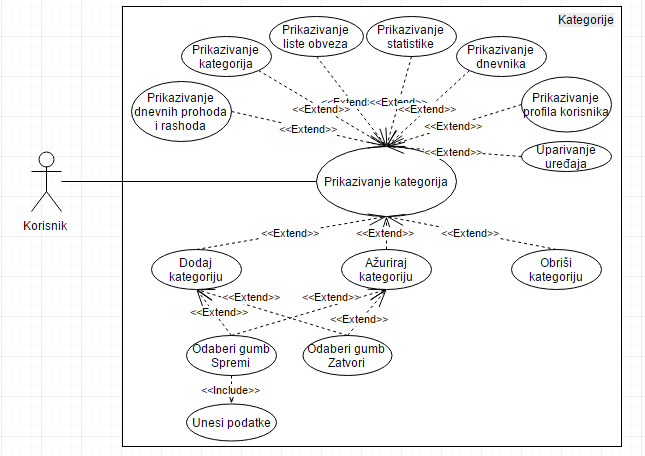
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-4 | Dnevni prihodi i rashodi |
| **Opis scenarija korištenja:**  Prikaz, unos, izmjena i brisanje prihoda i rashoda za određeni dan. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Prihodi/Rashodi aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Prikazuje se ekran Prihodi/Rashodi s popisom unesenih prihoda i rashoda, ukoliko istih ima. |
| 2 | Korisnik unosi novi prihod i/ili rashod i detalje o istom. |
| 3 | Korisnik može urediti postojeći prihod i/ili rashod. |
| 4 | Korisnik može obrisati postojeći prihod i/ili rashod. |



Slika 7: Dijagram slučaja korištenja Dnevni prihodi i rashodi

* + - 1. Slučaj korištenja Kategorije

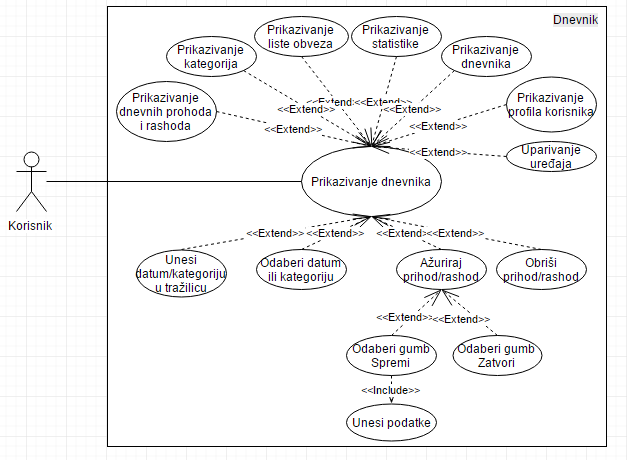
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-5 | Kategorije |
| **Opis scenarija korištenja:**  Prikaz, unos, izmjena i brisanje kategorija prihoda i rashoda. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Kategorije aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Kategorije iz navigacije. |
| 2 | Prikazuje se ekran Kategorije s popisom unesenih kategorija, ukoliko istih ima. |
| 3 | Korisnik odabire unos nove kategorije ukoliko želi dodati novu kategoriju prihoda ili rashoda. |
| 4 | Korisnik može urediti postojeću kategoriju. |
| 5 | Korisnik može obrisati postojeću kategoriju. |



Slika 8: Dijagram slučaja korištenja Kategorije

* + - 1. Slučaj korištenja Dnevnik

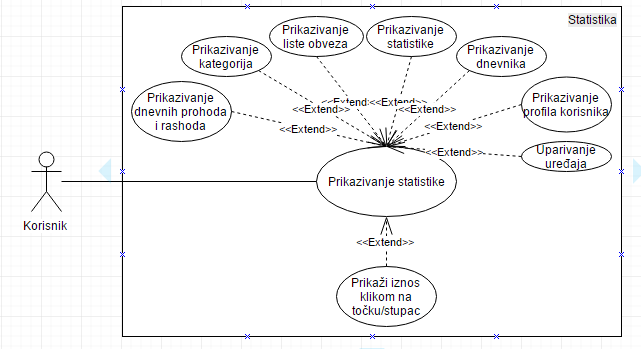
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-6 | Dnevnik |
| **Opis scenarija korištenja:**  Prikaz unesenih prihoda i rashoda unesenih na odabrani datum. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Dnevnik aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Dnevnik iz navigacije. |
| 2 | Prikazuje se ekran Dnevnik te mogućnost odabira prikaza prihoda i rashoda po datumu ili kategorijama. |
| 3 | Korisnik unosi datum ili naziv kategorije. |
| 4 | Prikazuju se prihodi i rashodi koje je korisnik unio za odabrani datum ili naziv kategorije. |



Slika 9: Dijagram slučaja korištenja Dnevnik

* + - 1. Slučaj korištenja Statistika

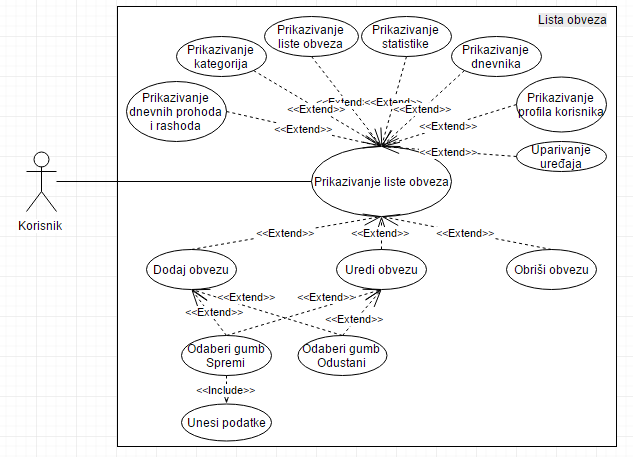
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-7 | Statistika |
| **Opis scenarija korištenja:**  Prikaz statistike korisnikovih prihoda i rashoda. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Statistika aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Statistika iz navigacije. |
| 2 | Prikazuje se ekran Statistika sa trenutnim financijskim stanjem unesenih prihoda i rashoda, te grafovima koji prikazuju statistike korisnikovih prihoda, statistiku rashoda, statistiku po kategorijama. |



Slika 10: Dijagram slučaja korištenja Statistika

* + - 1. Slučaj korištenja Lista obveza

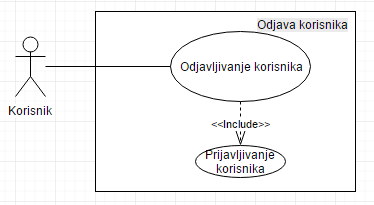
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-8 | Lista obveza |
| **Opis scenarija korištenja:**  Prikaz, unos, izmjena i brisanje obveza plaćanja na koje korisnika aplikacija pravovremeno upozorava. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Lista obveza aplikacije Personal Finance | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik odabire Lista obveza iz navigacije. |
| 2 | Prikazuje se ekran Lista obveza s postojećim obvezama u obliku datuma, naslova i opisa obveze te datuma kada korisnik želi obavijest od aplikacije za obvezu. |
| 3 | Korisnik unosi novu obvezu te uz nju potrebne podatke. |
| 4 | Korisnik može urediti postojeću obvezu. |
| 5 | Korisnik može obrisati postojeću obvezu. |



Slika 11: Dijagram slučaja korištenja Lista obveza

* + - 1. Slučaj korištenja Odjava korisnika

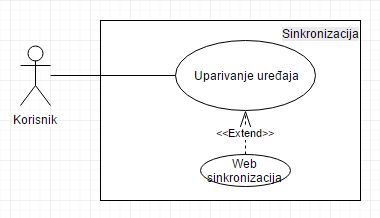
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-9 | Odjava korisnika |
| **Opis scenarija korištenja:**  Odjava korisnika iz aplikacije. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Moj profil aplikacije Personal Finance. | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik je na ekranu Moj profil. |
| 2 | Korisnik odabire opciju odjave iz aplikacije. |



Slika 12: Dijagram slučaja korištenja Odjava korisnika

* + - 1. Slučaj korištenja Sinkronizacija

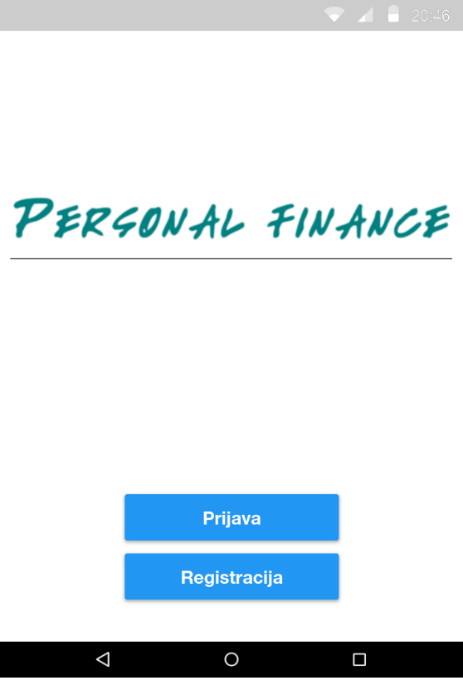
|  |  |
| --- | --- |
| **ID** | **Ime scenarija korištenja** |
| UC-10 | Sinkronizacija |
| **Opis scenarija korištenja:**  Web sinkronizacija podataka. | |
| **Preduvjeti:**   * Korisnik je prijavljen u aplikaciju * Android uređaj prikazuje ekran Upari uređaj aplikacije Personal Finance. | |
| **Korak** | **Opis koraka** |
| 1 | Korisnik je na ekranu Upari uređaj. |
| 2 | Korisnik odabire opciju Web sinkronizacija. |



Slika 13: Dijagram slučaja korištenja Sinkronizacija

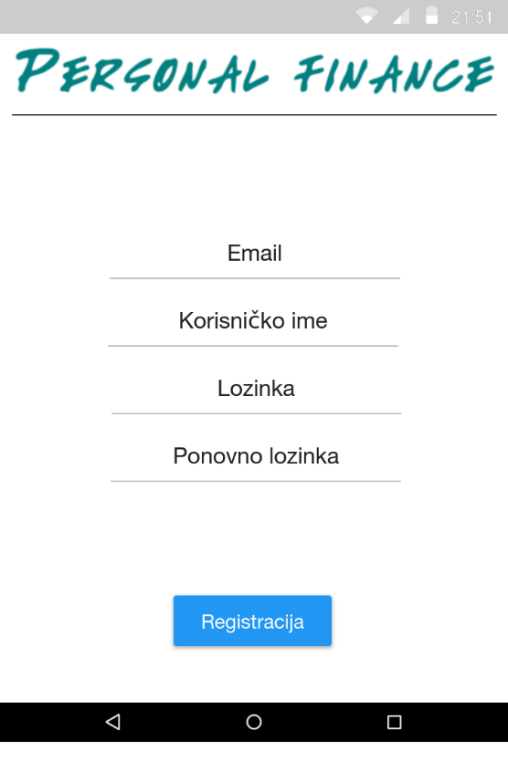
1. Mockup

Mockup je izrađen pomoću alata Proto.io, a prikazuje prototip aplikacije Personal Finance s finkcionalnostima i primarnim dizajnom aplikacije.



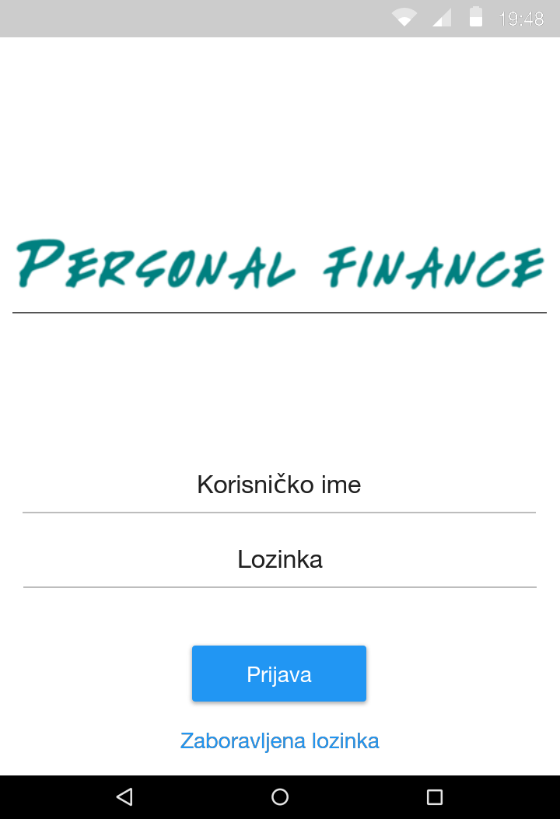
Slika 14: Početni prozor

Pokretanjem aplikacije otvara se početni prozor koji nudi dvije opcije. Ukoliko korisnik nema kreiran korisnički račun, odabire registraciju, a ukoliko ga ima, odabire prijavu. Ovisno o odabranom otvara se neki od sljedeća dva prozora.



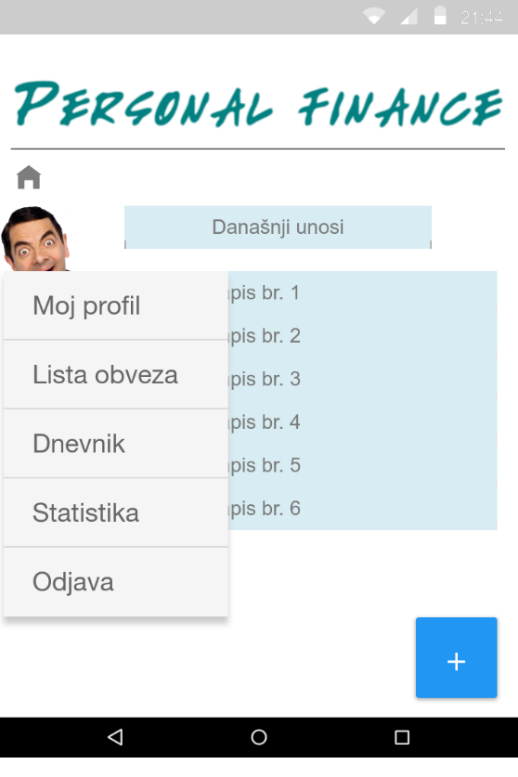
Slika 15: Registracija

Odabirom registracije, korisnik treba unijeti podatke potrebne za kreiranje korisničkog računa.



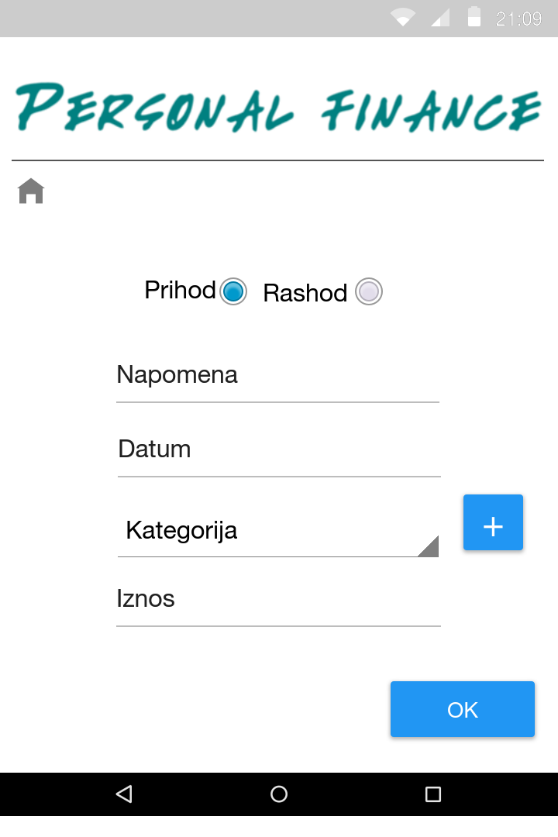
Slika 16: Prijava

Odabirom prijave, korisnik unosi podatke potrebne za otvaranje svog već postojećeg korisničkog računa.



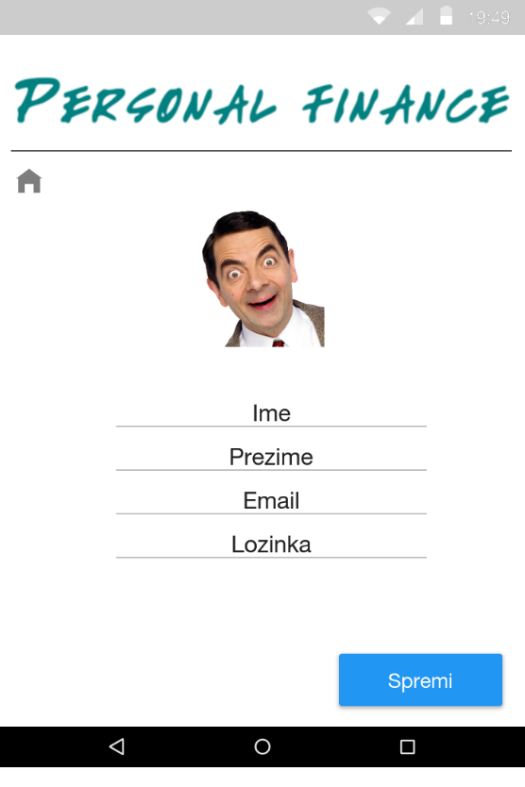
Slika 17: Navigacija i glavni prozor

Na prozoru je prikazana navigacija aplikacije koja sadrži opcije prikaza profila korisnika, liste obveza, dnevnika, statistike te odjavu korisnika iz aplikacije. Osim navigacije prikazan je i glavni prozor aplikacije, gdje korisnik vidi prihode i rashode od toga dana.



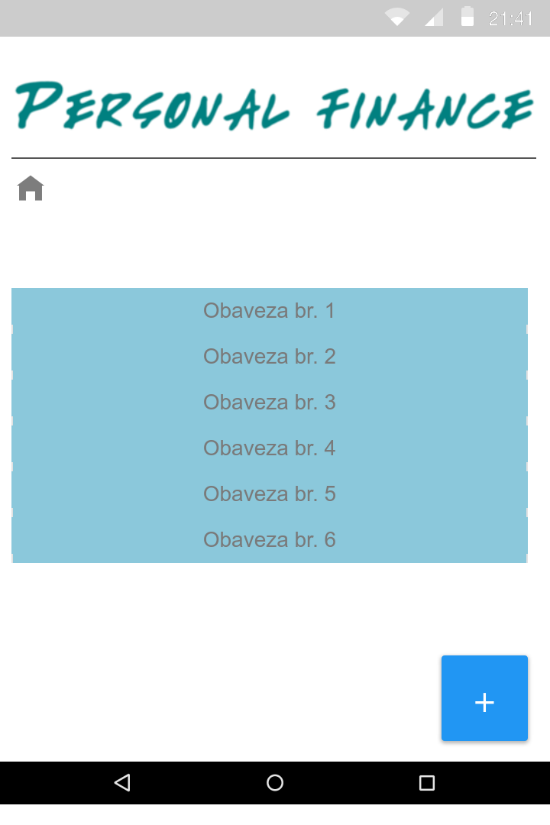
Slika 18: Novi zapis

Korisnik može dodati novi zapis, pri čemu određuje je li to prihod ili rashod, unosi iznos, bira kategoriju kojoj prihod/rashod pripada i sl.



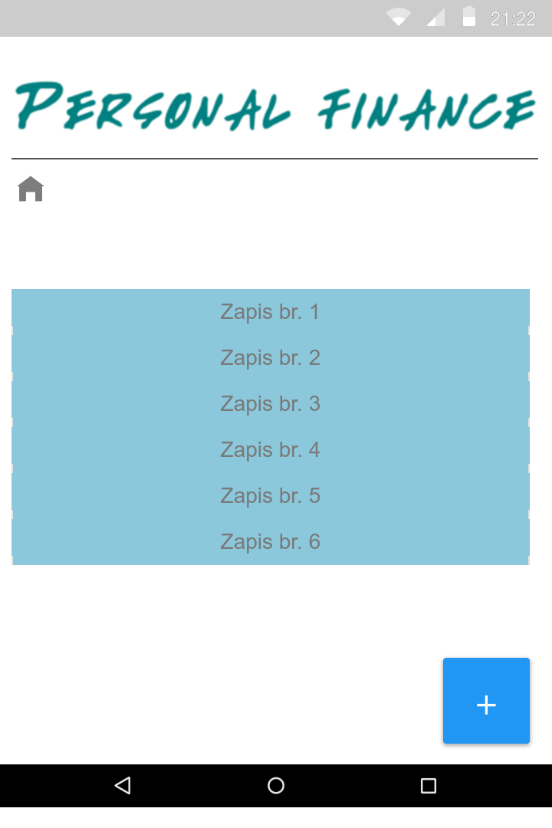
Slika 19: Profil

Odabirom profila korisniku se otvara prozor u kojem može unijeti i spremiti podatke o sebi.



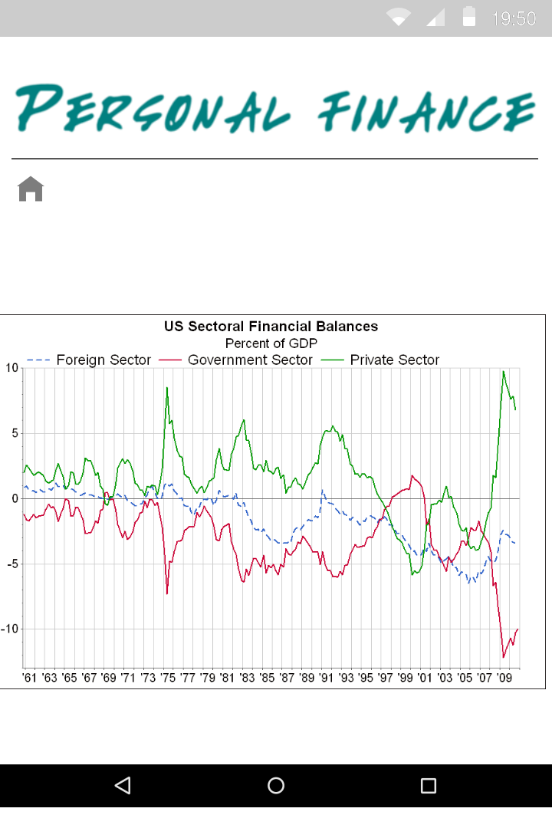
Slika 20: Lista obveza

Odabirom liste obveza korisniku se otvara prozor u kojem može vidjeti pregled trenutno unesenih obveza te dodati nove.



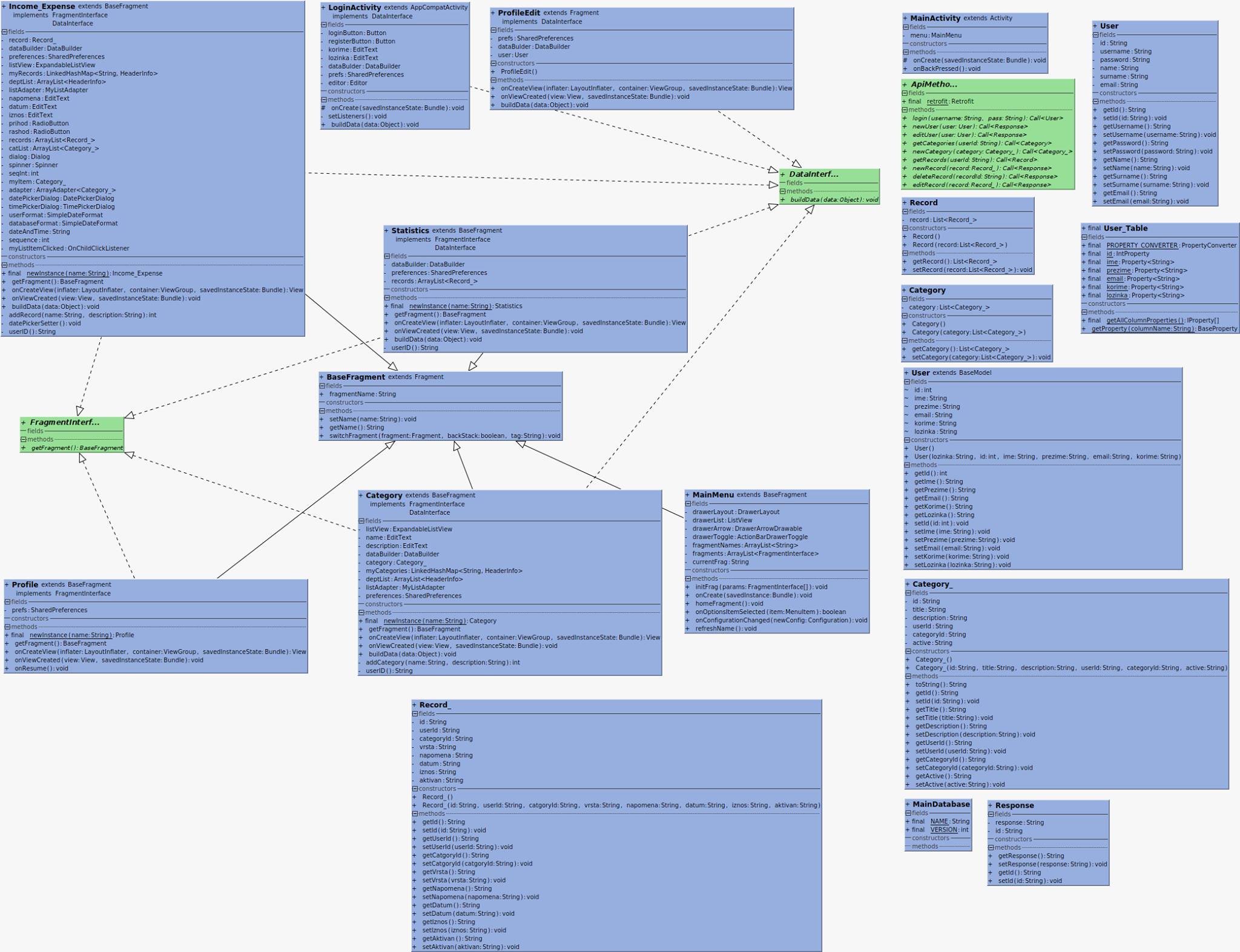
Slika 21: Dnevnik

Odabirom dnevnika korisniku se otvara prozor u kojem može vidjeti pregled zapisa po odabranom datumu.



Slika 22: Statistika

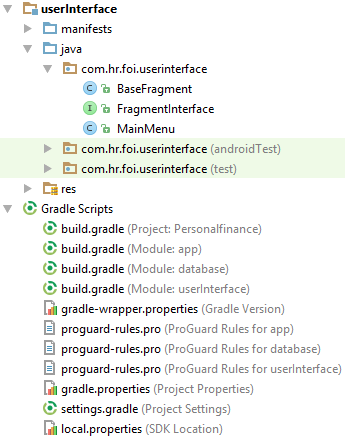
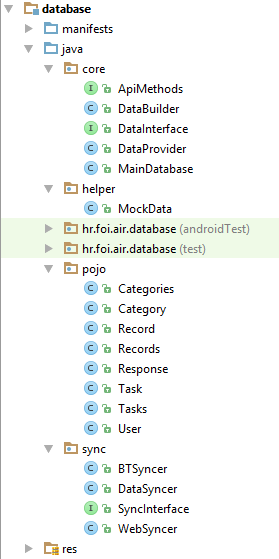
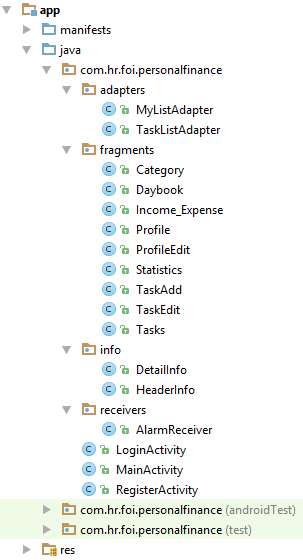
Odabirom statistike korisniku se otvara prozor u kojem može vidjeti statistički pregled svog financijskog kretanja u obliku grafa.

1. Dijagram klasa

Slika 23: Dijagram klasa

Dijagram klasa prikazuje klase generirane iz Android studia. Klase se sastoje od polja, konstruktora i metoda koje su izvučene iz programskog koda.

1. Struktura programskog koda



Slika 24: Prikaz strukture programskog koda

Slika 23 prikazuje strukturu programskog koda aplikacije Personal Finance iz Android Studio alata. Struktura se sastoji od tri glavna modula (app, database i userInterface) te od gradle-a. U app modulu nalaze se paketi adapters, fragments, info, receivers i dr. (struktura se vidi iz slike), tu su i aktivnosti aplikacije. Database modul sadrži pakete core, helper, pojo, sync i dr., dok userInterface modul služi za prikaz menu-a i upravljanje fragmentima, a nasljeđuje ga samo app modul.

1. Specifikacija web servisa

|  |  |
| --- | --- |
| **Title** | Registracija novog korisnika |
| **URL** | http://finance2016.000webhostapp.com/new\_user.php |
| **Method** | POST |
| **URL Params** | **Required:**  username=[string]  password=[string]  email=[string]  name=[string]  surname=[string] |
| **Data Params** | {  u : {  username : [string]  password : [string]  email : [string]  name : [string]  surname : [string]  }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Added“, id : 1}  **Code**:200  **Content**:{response : “Empty field “}  **Code**:200  **Content**:{response : “User exists “} |
| **Error Response** | **Code**:500  **Content**:{response : “Error INSERT INTO users VALUES (DEFAULT, 'username', 'password', 'email', 'name', 'surname');“} |
| **Sample Call** | @POST("/new\_user.php") Call<pojo.Response> newUser(@Body pojo.User user); |

Tablica 1. REST registracija novog korisnika

|  |  |
| --- | --- |
| **Title** | Prijava korisnika u aplikaciju |
| **URL** | http://finance2016.000webhostapp.com/ user\_by\_name.php?username&pass |
| **Method** | GET |
| **URL Params** | **Required:**  username=[string]  password=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { "id":"1","username":"admin","password":"admin", "email":"edit\_test", "name":"admin","surname":"ad" } |
| **Error Response** | **Code**:500  **Content**:{response : “Error“} |
| **Sample Call** | @GET("/user\_by\_name.php/") Call<pojo.User> login(@Query("username") String username, @Query("pass") String pass); |

Tablica 2. REST prijava korisnika

|  |  |
| --- | --- |
| **Title** | Ažuriranje informacija o korisniku |
| **URL** | http://finance2016.000webhostapp.com/edit\_user.php |
| **Method** | POST |
| **URL Params** | **Required:**  Id=[string]  username=[string]  password=[string]  email=[string]  name=[string]  surname=[string] |
| **Data Params** | {  u : {  id : [string]  username : [string]  password : [string]  email : [string]  name : [string]  surname : [string]  }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Promijenjeno“, id : 1}  **Code**:200  **Content**:{response : “Prazno polje “} |
| **Error Response** | **Code**:500  **Content**: {response : “Greška!“, id : 2} |
| **Sample Call** | @POST("/edit\_user.php") Call<pojo.Response> editUser(@Body pojo.User user); |

Tablica 3. REST ažuriranje informacija o korisniku

|  |  |
| --- | --- |
| **Title** | Dohvaćanje kategorija korisnika |
| **URL** | http://finance2016.000webhostapp.com/ category\_by\_user.php?id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { "category":[{"id":"75","title": "nova12", "description":"opis2","lastEdited":"2017-02-13 02:54:05"}]} |
| **Error Response** | None |
| **Sample Call** | @GET("/category\_by\_user.php/") Call<Categories> getCategories(@Query("id") String userId); |

Tablica 4. REST dohvaćanje kategorija

|  |  |
| --- | --- |
| **Title** | Dodavanje nove kategorije |
| **URL** | http://finance2016.000webhostapp.com/add\_category.php |
| **Method** | POST |
| **URL Params** | **Required:**  user\_id=[string]  title=[string]  description=[string] |
| **Data Params** | {  u : {  user\_id: [string]  title: [string]  description: [string]  }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty field! “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “SQL Error. Query: INSERT INTO category VALUES (DEFAULT, 'title', 'description');“, id : -1} |
| **Sample Call** | @POST("/add\_category.php") Call<pojo.Response> newCategory(@Body Category category); |

Tablica 5. REST dodavanje nove kategorije

|  |  |
| --- | --- |
| **Title** | Dohvaćanje zapisa po korisniku |
| **URL** | http://finance2016.000webhostapp.com/ records\_by\_user.php?id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { " record":[{"id":"14","user\_id":"1","category\_id":null,"vrsta":"0","napomena":"pp1","datum":"2017-01-30 00:00:00", "iznos":"1288.00" ,"aktivan":"1","lastEdited":"2017-02-13 02:56:40"}]} |
| **Error Response** | None |
| **Sample Call** | @GET("/records\_by\_user.php/") Call<Records> getRecords(@Query("id") String userId); |

Tablica 6. REST dohvaćanje zapisa po korisniku

|  |  |
| --- | --- |
| **Title** | Dodavanje novog zapisa |
| **URL** | http://finance2016.000webhostapp.com/new\_record.php |
| **Method** | POST |
| **URL Params** | **Required:**  user\_id=[string]  vrsta=[string]  napomena=[string]  datum=[string]  iznos=[string]  aktivan=[string] |
| **Optional:**  category\_id=[string] |
| **Data Params** | {  u : {  user\_id: [string]  category\_id : [string]  vrsta : [string]  napomena : [string]  datum : [string]  iznos : [string]  aktivan : [string] }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty field! “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “Unknown database“, id : -1} |
| **Sample Call** | @POST("/new\_record.php") Call<pojo.Response> newRecord(@Body Record record); |

Tablica 7. REST dodavanje novog zapisa

|  |  |
| --- | --- |
| **Title** | Brisanje zapisa |
| **URL** | http://finance2016.000webhostapp.com/ delete\_record.php?id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { response: “Succes“, id : 1}  **Code**: 200  **Content**: { response: “Empty id“, id : -2} |
| **Error Response** | **Code**: 500  **Content**: { response: “Error“, id : -1} |
| **Sample Call** | @GET("/delete\_record.php") Call<pojo.Response> deleteRecord(@Query("id") String recordId); |

Tablica 8. REST Brisanje zapisa

|  |  |
| --- | --- |
| **Title** | Ažuriranje informacija o zapisu |
| **URL** | http://finance2016.000webhostapp.com/edit\_record.php |
| **Method** | POST |
| **URL Params** | **Required:**  user\_id=[string]  vrsta=[string]  napomena=[string]  datum=[string]  iznos=[string] |
| **Optional:**  category\_id=[string] |
| **Data Params** | {  u : {  user\_id: [string]  category\_id : [string]  vrsta : [string]  napomena : [string]  datum : [string]  iznos : [string] }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty id “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “Unknown database“, id : -1} |
| **Sample Call** | @POST("/edit\_record.php") Call<pojo.Response> editRecord(@Body Record record); |

Tablica 9. REST Ažuriranje zapisa

|  |  |
| --- | --- |
| **Title** | Dodavanje nove obaveze |
| **URL** | http://finance2016.000webhostapp.com/new\_task.php |
| **Method** | POST |
| **URL Params** | **Required:**  user\_id=[string]  title=[string]  note=[string]  date=[string]  notice=[string] |
| **Data Params** | {  u : {  user\_id: [string]  title : [string]  note : [string]  date: [string]  notice: [string] }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty field “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “Unknown database“, id : -1} |
| **Sample Call** | @POST("/new\_task.php") Call<pojo.Response> newTask(@Body Task task); |

Tablica 10. REST dodavanje nove obaveze

|  |  |
| --- | --- |
| **Title** | Dohvaćanje obaveza za korisnika |
| **URL** | http://finance2016.000webhostapp.com/ tasks\_by\_user.php?id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { tasks":[{"id":"1","user\_id":"1","title":"test","note":"edit test","date":"2017-01-18 04:00:00","notice":"2017-01-18 03:00:00" ,"aktivan":"1","lastEdited":"2017-02-13 02:56:55"}]} |
| **Error Response** | None |
| **Sample Call** | @GET("/tasks\_by\_user.php") Call<Tasks> getTasks(@Query("id") String userId); |

Tablica 11. REST dohvaćanje obaveza za korisnika

|  |  |
| --- | --- |
| **Title** | Brisanje obaveza za korisnika |
| **URL** | http://finance2016.000webhostapp.com/ delete\_task.php?id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { response: “Succes“, id : 1}  **Code**: 200  **Content**: { response: “Empty id“, id : -2} |
| **Error Response** | **Code**: 500  **Content**: { response: “Error“, id : -1} |
| **Sample Call** | @GET("/delete\_task.php") Call<pojo.Response> deleteTask(@Query("id") String id); |

Tablica 12. REST brisanje obaveza

|  |  |
| --- | --- |
| **Title** | Ažuriranje obaveze |
| **URL** | http://finance2016.000webhostapp.com/edit\_task.php |
| **Method** | POST |
| **URL Params** | **Required:**  id=[string]  title=[string]  note=[string]  date=[string]  notice=[string] |
| **Data Params** | {  u : {  id: [string]  title : [string]  note : [string]  date: [string]  notice: [string] }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty id “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “Unknown database“, id : -1} |
| **Sample Call** | @POST("/edit\_task.php") Call<pojo.Response> editTask(@Body Task task); |

Tablica 13. REST ažuriranje obaveze

|  |  |
| --- | --- |
| **Title** | Ažuriranje kategorije |
| **URL** | http://finance2016.000webhostapp.com/edit\_ category..php |
| **Method** | POST |
| **URL Params** | **Required:**  id=[string]  title=[string]  description=[string] |
| **Data Params** | {  u : {  id: [string]  title : [string]  description : [string] }  } |
| **Sucess Response** | **Code**: 200  **Content**: {response : “Success“, id : 1}  **Code**:200  **Content**:{response : “Empty id “, id : -2} |
| **Error Response** | **Code**:500  **Content**: {response : “Unknown database“, id : -1} |
| **Sample Call** | @POST("/edit\_category.php") Call<pojo.Response> editCategory(@Body Category category); |

Tablica 14. REST ažuriranje obaveze

|  |  |
| --- | --- |
| **Title** | Brisanje kategorije |
| **URL** | http://finance2016.000webhostapp.com/ delete\_task.php?id&user\_id |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string]  user\_id=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: { response: “Succes“, id : 1}  **Code**: 200  **Content**: { response: “Empty id“, id : -2} |
| **Error Response** | **Code**: 500  **Content**: { response: “Error“, id : -1} |
| **Sample Call** | @GET("/delete\_category.php") Call<pojo.Response> deleteCategory(@Query("id") String id ,@Query("user\_id") String user\_id); |

Tablica 15. REST brisanje kategorije

|  |  |
| --- | --- |
| **Title** | Dohvaćanje posljednje izmijenjenih kategorija |
| **URL** | http://finance2016.000webhostapp.com/ getEditedCategories.php?id&lastEdited |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string]  lastEdited=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: {"category":[{"id":"1","title":"naslov","description":"opis","lastEdited": "2017-02-13 02:54:15"}]}  **Code**: 200  **Content**: { response: “Empty last edited value“, id : -2}  **Code**: 200  **Content**: { response: “Empty ID“, id : -1} |
| **Error Response** | None |
| **Sample Call** | @GET("/getEditedCategories.php") Call<pojo.Categories> getEditedCategories(@Query("id") String id, @Query("lastEdited") String lastEdited); |

Tablica 16. REST dohvaćanje posljednje izmijenjenih kategorija

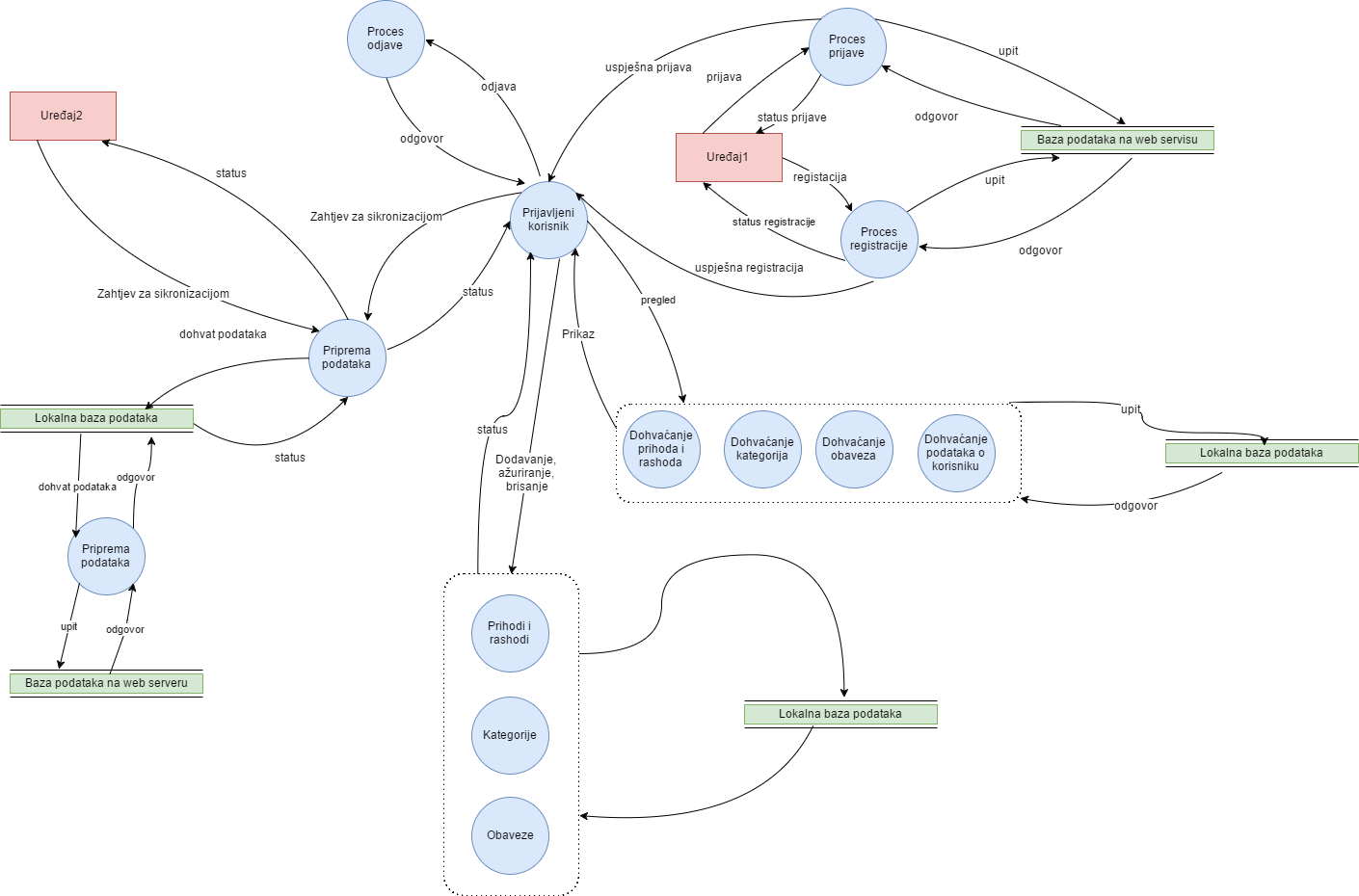
|  |  |
| --- | --- |
| **Title** | Dohvaćanje posljednje izmijenjenih zapisa |
| **URL** | http://finance2016.000webhostapp.com/ getEditedRecords.php?id&lastEdited |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string]  lastEdited=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: {" record":[{"id":"14","user\_id":"1","category\_id":null,"vrsta":"1", "napomena":"pp","datum":"2017-01-30 00:00:00", "iznos":"1288.00","aktivan":"1","lastEdited":"2017-02-13 02:56:40"}]}  **Code**: 200  **Content**: { response: “Empty last edited value“, id : -2}  **Code**: 200  **Content**: { response: “Empty ID“, id : -1} |
| **Error Response** | None |
| **Sample Call** | @GET("/getEditedRecords.php") Call<pojo.Records> getEditedRecords(@Query("id") String id, @Query("lastEdited") String lastEdited); |

Tablica 17. REST dohvaćanje posljednje izmijenjenih zapisa

|  |  |
| --- | --- |
| **Title** | Dohvaćanje posljednje izmijenjenih obaveza |
| **URL** | http://finance2016.000webhostapp.com/ getEditedTasks.php?id&lastEdited |
| **Method** | GET |
| **URL Params** | **Required:**  id=[string]  lastEdited=[string] |
| **Data Params** | None |
| **Sucess Response** | **Code**: 200  **Content**: {" tasks":[{"id":"1","user\_id":"1","title":"test","note": "edit test","date":"2017-01-18 04:00:00","notice":"2017-01-18 03:00:00","aktivan":"1","lastEdited":"2017-02-13 02:56:55"}]}  **Code**: 200  **Content**: { response: “Empty last edited value“, id : -2}  **Code**: 200  **Content**: { response: “Empty ID“, id : -1} |
| **Error Response** | None |
| **Sample Call** | @GET("/getEditedTasks.php") Call<pojo.Tasks> getEditedTasks(@Query("id") String id, @Query("lastEdited") String lastEdited); |

Tablica 18. REST dohvaćanje posljednje izmijenjenih obaveza

1. Pristup u radu s podacima

****

Slika 25: Dijagram toka podataka

Dijagram prikazuje tok podataka aplikacije Personal Finance.

1. Testiranje
   1. Uputstvo za testiranje aplikacije

Prilikom pokretanja aplikacije otvara se prozor za prijavu, ukoliko korisnik ima kreiran korisnički račun, unosi korisničko ime i lozinku (po default-u: admin admin). Ukoliko nema račun, klikom na gumb Registracija, aplikacija ga vodi na prozor za registraciju gdje je potrebno unijeti nekoliko podataka kako bi se kreirao korisnički račun. Korisnik je nakon toga prijavljen u aplikaciju. Otvara mu se glavni prozor aplikacije s prikazanim prihodima i rashodima ukoliko ih je unio, ukoliko nije, pritiskom na gumb Dodaj, dodaje nove prihode/rashode i njihove detalje. U navigation drawer-u korisnik može odabrati Kategorije – gdje može dodavati nove kategorije po vlastitom izboru; Lista obveza – gdje vidi listu obveza ukoliko je unio obveze, klikom na gumb može dodati novu, a aplikacija ga obaviještava na obvezu u dogovoreno vrijeme; Statistika – gdje vidi nekoliko grafova koji prikazuju prihode i/ili rashode; Dnevnik – gdje se korisniku odabire datum za koji želi vidjeti unesene prihode/rashode; Moj profil – gdje vidi osobne podatke koji su iz baze povučeni prilikom registracije, pritiskom ikone za uređivanje, može iste podatke i promijeniti, dok pritiskom ikone za odjavljivanje, može se odjaviti iz aplikacije.

* 1. UI testovi

Za izradu testova ovog projekta, koristen je library Robotium[[1]](#footnote-1). To je vrlo popularan open-source Android framework za pisanje automatiziranih testova. Moć Robotiuma leži u tome da pored gray box testiranja omogućava i black box testiranje kada izvorni kod nije dostupan nego samo njegova .apk datoteka.

Korištena je najnovija verzija (5.6.3). Framework se u projekt dodaje kao i svaki vanjski library, npr. putem Gradle konfiguracijske datoteke.

Napravljeni su jednostavni testovi za sljedeće aktivnosti: LoginActivity i RegisterActivity. U oba testa, provjerava se jesu li unesene vrijednosti null nakon onClick() događaja nad gumbima login i register.

1. <https://github.com/RobotiumTech/robotium> [↑](#footnote-ref-1)