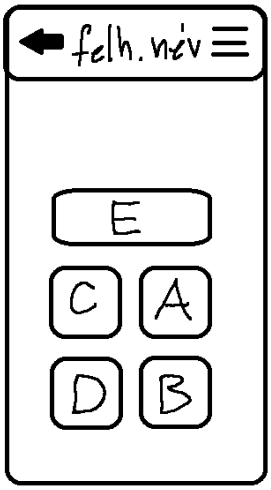
Önálló Laboratórium

Alkalmazás specifikáció

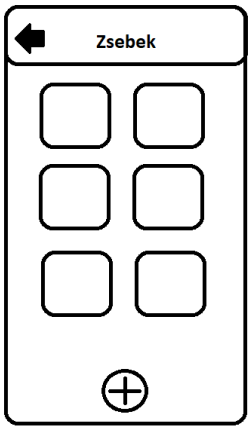
Kezdőképernyő, bejelentkezés:

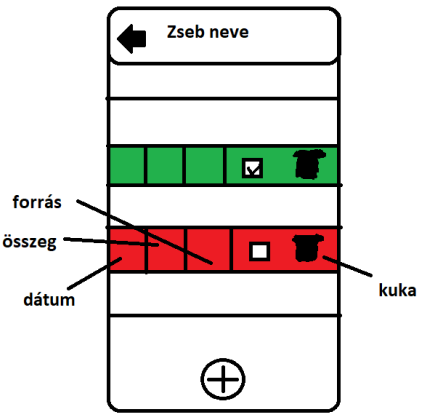
- ha a felhasználó még nincs beregisztrálva, vagy nincsenek kitöltve megfelelően a mezők, akkor egy hibának megfelelő Toastot dob az alkalmazás.
- A regisztrációra kattintva lehet regisztrálni egy új felhasználót az alkalmazásba.
- Belépés után a jobboldali képernyő fogad.

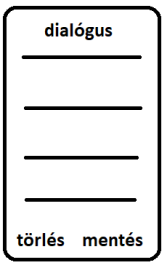


Funkció A, B:

Az alkalmazásban lehet vezetni a kiadásokat és a bevételeket a következők mellett: **dátum**, **kiadás/bevétel forrása**, **megjegyzések**. A megjegyzés nem kötelező mező. Ezeket a rekordokat az alkalmazás elmenti egy adatbázisba a telefon lokális tárhelyére, illetve egy szerverre. Különböző kiadásoknak/bevételeknek zsebeket lehet létrehozni, ahol a tranzakció végbemegy. Az A vagy a B funkcióra rányomva előbb a zsebek jelennek meg, ahol ki lehet választani, hogy melyik zsebből szeretnénk költeni, vagy éppen hozzáadni. Új zsebet is itt lehet létrehozni. A dátum mező automatikusan kitöltődik, ha máshogy nem rendelkezünk.





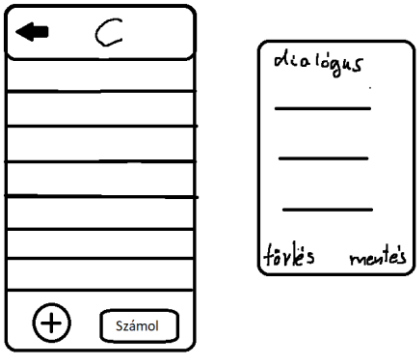


A + gombra rányomva lehet új rekordot létrehozni. Ekkor egy dialógus ablak ugrik fel, aminek kitöltésével már kész is az új rekord.

- Egy régebbi rekordra ha rákattint a felhasználó, akkor is előugrik a dialógusablak, ekkor módosítani, vagy törölni lehet a meglévő rekordot.
- A vissza gomb visszavisz az előző oldalra.

Funkció C:

Az alkalmazásba fel lehet tölteni különböző tartozásokat nevekkel együtt, hogy lehessen látni, ki tartozik kinek és mennyivel. Itt is van egy megjegyzés mező, ami opcionális. Egy gombnyomással az app összegezni tudja a tartozásokat, és megmondja, a legkevesebb olyan tartozásokat, melyekkel mindenki csak a sajátját tételeiért fizet.



- A + gombbal megnyílik egy dialógusablak, aminek kitöltésével létrejön egy új rekord a tartozások között.
- A Számol gomb összegzi a tartozásokat és leredukálja a lehető legkevesebb tranzakcióra.
- A kifizetett tranzakciókat egy checkbox bepipálásával lehet jelezni, ami után a rekord view-jának a színe megváltozik pirosra, vagy zöldre.

Funkció D:

Itt lehet vezetni a befektetéseket. A felhasználó valamelyik zsebből le tud írni egy általa választott összeget, és hozzá ír egy kommentet/megjegyzést, hogy mibe fektet be pl: tőzsde, kötvény, vagy kriptodeviza. Ennél a fülnél egy listában lehet látni a különböző befektetéseket és hozzá lehet írni az aktuális értékét Ft-ban, amihez egy időbélyeg is társul, így lehet látni, hogy mikor mennyit ért a befektetés, ami később statisztikai célokra is jó lehet.

- Az előző funkcióhoz hasonlóan itt is megjelenik minden rekordhoz egy megfelelő dialógusablak, a mezőkből előre kitöltve a rekordban lévő információkkal. Ezáltal könnyen lehet módosítani a rekordot, illetve törölni is lehet.

Funkció E:

Itt rá lehet szűrni, különböző zsebek tartalmára. Egy listában kiválasztja a felhasználó, hogy mely zsebeket szeretné a szűrésben látni, majd a dátumra és a forrásra rákereshet. Minden keresésnek megfelelt találatot kilistáz az alkalmazás. A szűrésnél a * karakter is használható.

- A megfelelő zseb kiválasztása után jelenik meg a baloldali képernyő
- A szűrt rekordok a fentiekhez hasonlóan egy RecycleView-ban jelennek meg.
- A listázott rekordok végösszege a felső sávban jelenik meg.

Fejlesztői Specifikáció

Az alkalmazás fejlesztése során az MVVM mintát és a három rétegű architektúrát követem. Perzisztens tárolásra a Firebase-nek az ingyenes szerver szolgáltatását használom, aminek segítségével egy vastagkliens alkalmazást hozok létre, tehát az üzleti logika a kliensoldalon lesz megtalálható.

View:

A fentebb leírt funkcióleírások mellett láthatóak a view-k:

- loginActivity
- signUpActivity
- mainMenuActivity
- pocketActivity
- spendActyvity
- earnActivity
- investActivity
- loanActivity
- queriesActivity
- + a megfelelő dialógus ablakok

Feladatuk: a UI elemeket tárolják

ViewModel:

- AuthViewModel
  - feladata: a felhasználó autentikálása
- spendViewModel
  - feladatai: egy új rekord létrehozása az adatbázisba
  - dialógus ablak kezelése
- earnViewModel
  - feladatai: egy új rekord létrehozása az adatbázisba

    - dialógus ablak kezelése
- loanViewModel
    - feladatai: egy új rekord létrehozása az adatbázisba
    - dialógus ablak kezelése
    - kiszámolja, hogy legkevesebb mely tranzakciók elvégzésével jöhet ki mindenki 0-ra.
- investViewModel
    - feladatai: egy új rekord létrehozása az adatbázisba
    - dialógus ablak kezelése
- queryViewModel
    - feladata: szűrés és számolás elvégzése
- networkViewModel
    - feladata: a szerverrel való kommunikálás a hálózaton

Model:

A következő Repositori-k vannak:

- SpendingRepo: egy kiadás/rekord információit tárolja
- EarningRepo: egy bevétel/rekord információit tárolja
- InvestingRepo: egy befektetés/rekord információit tárolja
- IntervalSumRepo: egy befektetéshez tartozó részösszegeket tárolja
- LoanRepo: egy tartozás adatait tárolja
- PocketRepo: A zsebeket tárolja

A lokális mentéshez Room adatbázist használok, amihez az SQLite lekérdezőnyelvet használom. 6 táblám lesz:

Spending tábla mezői:

1. ID: egyedi kulcs
2. PocketID: zseb ID-ja (külső kulcs)
3. dátum: év.hónap.nap
4. bevétel
5. bevétel forrása
6. megjegyzés

Earning tábla mezői:

1. ID: egyedi kulcs
2. PocketID: zseb ID-ja (külső kulcs)
3. dátum: év.hónap.nap
4. kiadás
5. kiadás forrása
6. megjegyzés

Invest tábla mezői:

1. ID: egyedi kulcs
2. összeg
3. hova ment

4. komment

IntervalSum tábla mezői:

1. ID: egyedi kulcs
2. InvestID: melyik Invest táblabeli rekordhoz tartozik (külső kulcs)
3. összeg
4. komment

Pocket tábla mezői:

1. ID: egyedi kulcs
2. zseb neve
3. zsebben lévő összeg

Loan tábla mezői:

1. ID: egyedi kulcs
2. ki: ki tartozik (egyedi név)
3. kinek: kinek tartozik (egyedi név)
4. leírás: a tartozás oka
5. összeg: a tartozás összege
6. ki van-e fizetve