**Kiadás követő alkalmazás**

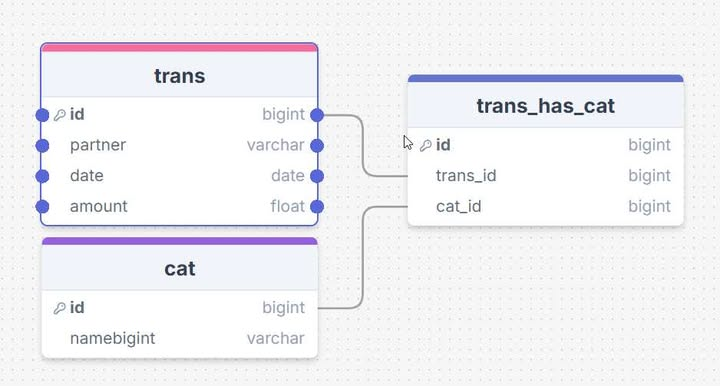
Leírás:

Segít a felhasználónak rendszerezni és számontartani a pénzügyeit.

**Kozma Kristóf**

háttérben: sqlite python

funckciók: add\_trans(amount:float, partner:str, date:date, cats:list): Returns: Trans object get\_all\_trans() Returns: List[Trans objects]  
add\_cat(name:str) Returns: Trans object  
get\_all\_cat() Returns: List[Kat objects]  
add\_cat\_to\_trans(trans,cat)



Molnár Csegő Márk

Látványterv



**Menüpontok:**

**Molnár Csegő**

**Kiadások/Bevétel-egyszeri:**

* Leírás:

A felhasználó itt adhatja meg milyen egyszeri kiadások és bevételek érték az adott időben továbbá kiírja egy json fileba hogy mennyi pénze van

* Függvények:

BevetelKiadas(partner, amount, balance, kapcs)  
Az összes adatot a fülön a gombokból szedi ki  
Hozzáadja vagy kivonja a kapcs checkbox vagy +- érték alapján  
Elmenti a balance-t a fileba Hozzáadja az adatbázishoz minden adatot

Belvasas()   
Visszatér egy json fileból beolvasott balance-al

Kiiras(balance)  
Beleírja a json fileba a balance-ot

**Botos Etele**

**Kiadások/Bevétel-rendszeres:**

* Leírás:

A felhasználó a programban meg tud adni bevételt vagy kiadást. A hozzáadásnál megadható az összeg, kategória, dátum, valamint a gyakoriság (napi, heti, havi, éves vagy egyedi).

* Függvények:

hozzaad\_rendszeres(osszeg, leiras, kategoria, gyakorisag, kezdet\_datum)  
Új rendszeres bevétel vagy kiadás hozzáadasa a listához a megadott adatokkal.

elo\_fordulasok(kezdet, veg)   
 Megnézi, hogy egy rendszeres tétel hányszor fordul elő egy adott időszakban.

frissit\_tranzakciokat()  
Frissíti a képernyőn látható listát, hogy az új bevétel vagy kiádas azonnal megjelenjen.

ment\_tranzakcio()   
A mentes gomb működéséért felel, elmenti az adatokat és frissíti a felületet.

**Dajka Bence Imre**

Befektetések:

**Vaskó Dániel**

**Statisztika**

* Leírás:

Az alkalmazás statisztikai modulja a felhasználó költési szokásait elemzi és vizuálisan is megjeleníti. A cél, hogy a felhasználó átlátható képet kapjon a kiadásairól, kategóriák szerinti megoszlásáról és azok időbeli alakulásáról.

* Függvények:

-Összes költség:  
 Az eddigi kiadásokat mutatja meg. - total\_expense(list[Trans objects]) -> int

-Kategóriénkénti csoportosítás:  
 A költségek kategóriák szerinti összesítését kördiagram formájában ábrázolja, így gyorsan látható, mely területeken költ legtöbbet a felhasználó (pl. étkezés, közlekedés, szórakozás). - total\_by\_category() -> list[tuple[str, float]]: -Havi statisztika a költségekről, időszerinti lebontásban: A kiadásokat időrendben mutatja be hisztogram segítségével. - monthly\_expenses(year: int, month: int) -> list[tuple[str, float]] -Átlagos kiadás: Heti és havi bontásban számítja ki az átlagos költést. - average\_expense(List[Trans objects]) -> float

-(opcionális) Szórás:  
 Amennyiben elegendő adat áll rendelkezésre, a szórás segítségével mérhető, mennyire kiegyensúlyozott a felhasználó költségei. std\_expense()

**Mázsári Patrik**

**Félretétel:**

* Leírás

A felhasználó a programban meg tud adni egy összeget, amit félre szeretne tenni.

A rendszer folyamatosan figyeli a kiadásokat és bevételeket, és ha a rendelkezésre álló pénz (aktuális egyenleg) a félretett összeg alá csökken, akkor figyelmeztetést jelez.

* Függvények

beallit\_felretett(osszeg)  
Beállítja, mennyi pénzt szeretne a felhasználó félretenni.  
Az adatot elmenti az adatbázisba vagy egy változóba.

ellenoriz\_keret(aktualis\_egyenleg, felretett\_osszeg)  
Megvizsgálja, hogy az aktuális egyenleg nagyobb-e vagy kisebb a félretett összegnél.  
Ha az egyenleg kisebb, figyelmeztető üzenetet jelenít meg („Túllépted a félretett pénzt!”).

jelzes\_uzenet(szoveg, tipus) Kezeli a vizuális jelzéseket.  
 • Ha a típus „figyelmeztetés”, piros színű üzenetet vagy felugró ablakot jelenít meg.  
 • Ha minden rendben, zöld visszajelzést ad („Minden rendben, a kereten belül vagy.”).

frissit\_egyenleg(uj\_egyenleg)  
Frissíti az aktuális egyenleget a legutóbbi tranzakciók alapján, és automatikusan újraellenőrzi, hogy túllépés történt-e.

ment\_felretett()  
A „Mentés” gomb működéséért felel. Elmenti a beállított félretett összeget és frissíti a felületet az aktuális állapotnak megfelelően.

**Varga Gábor Áron:**

Múltbéli tranzakciók:

* Leírás:

A felhasználó a keresési felületen különböző szűrőfeltételeket állíthat be (pl. kategória, dátum intervallum, gyakoriság, összeg tartomány, leírás). A program a megadott feltételek alapján megjeleníti azokat a tranzakciókat, amelyek megfelelnek a keresési kritériumoknak. A szűrés valós időben vagy a „Keresés” gomb megnyomásával történik, és a találatok azonnal frissülnek a képernyőn.

* Függvények:

Keresés kategória alapján  
Csak az adott kategóriához tartozó bevételek vagy kiadások jelennek meg.

Keresés dátum intervallum szerint:  
A felhasználó megadhat egy kezdő- és egy végdátumot, és a rendszer csak az ebbe az időszakba eső tételeket listázza.

Keresés összeg alapján(trans lista)  
Opcionálisan megadható minimum és maximum összeg, amely alapján a program kiszűri a megfelelő tételeket.

Keresés gyakoriság alapján(trans lista)  
Lehetőség van a napi, heti, havi, éves vagy egyedi típusú rendszeres tételek szűrésére.

Szöveges keresés(trans lista)  
A leírás mezőben szereplő kulcsszavak alapján is lehet keresni (pl. „fizetés”, „villanyszámla” stb.).

**Kiss Péter**

**Kategóriák:**

* Leírás:

A felhasználó kategóriákat tud létrehozni, módosítani és törölni. A kategóriák listája egy legördülő menüben jelenik meg, amelyből a felhasználó kiadás vagy bevétel rögzítésekor kiválaszthatja a megfelelő kategóriát. A kategóriák egy külső fájlban .json vagy .txt kerülnek mentésre, és a program indításakor automatikusan betöltődnek. A felhasználó új kategóriát adhat hozzá egy + gombbal meglévőt módosíthat vagy törölhet.

* Függvények:

create\_category(name: str)  
Új kategóriát hoz létre a listában, ha még nem létezik

rename\_category(old\_name: str, new\_name: str)  
Megváltoztatja egy létező kategória nevét delete\_category(name: str) Eltávolít egy kategóriát a listából

list\_categories()  
Visszaadja az összes kategóriát listaként

save\_to\_file(filepath: str)  
Elmenti a kategóriákat egy fájlba

load\_from\_file(filepath: str)  
Betölti a kategóriákat egy fájlból

**Dajka Bence Imre:**

Befektetések:

A befektetési modul célja, hogy a felhasználók pénzügyi vagyonukat különböző kockázatú és időtávú befektetésekbe helyezhessék el. A modul segít megérteni a befektetési döntések várható hozamát és kockázatát, valamint előre kalkulálja a várható eredményt adott befektetett összeg, időtáv és modell alapján.

A felhasználó három alapvető befektetési modellt választhat:

* **Alacsony kockázatú modell:** Elsősorban államkötvények, stabil portfólióelemek. Lassú, de biztos növekedés.
* **Közepes kockázatú modell:** Vegyes alapú portfólió részvényekkel, kötvényekkel és ETF-ekkel.
* **Magas kockázatú modell:** Főként részvények és kriptovaluták dominálta befektetés. Nagyobb hozam lehetősége, de jelentősebb kockázattal.

Ezen túl a felhasználó választhat befektetési **időtávot** is (rövid, közép, hosszú táv), amelyek az elvárt hozam és kockázat figyelembevételével módosítják a kalkulált eredményt.

A modul egyik fő funkciója a **hozamszámítás**, amely a következőképp működik:  
A felhasználó megad egy tetszőleges befektetési összeget, majd a rendszer kalkulálja a várható hozamot az éves kamatláb és kockázati profil alapján

**Függvények:**

* valaszdmodell(kockazati\_szint: str)  
  Beállítja a kiválasztott kockázati modellt (alacsony, közepes, magas).
* valaszdidotav(idotav: str)  
  Beállítja a befektetési időhorizontot (rövid, közép, hosszú).
* szamithozam(osszeg: float, modell: str, idotav: str)  
  Visszaadja a várható hozamot az adott paraméterek alapján, százalékos és abszolút értékben.

**Felhasználói felület:**  
A modul külön fülön érhető el az alkalmazásban. A felhasználó itt megadhatja a kezdő összeget, kiválaszthatja a modell típusát és időtávját, majd egy „Kalkuláció” gomb megnyomásával megtekintheti a várható hozamot és annak időbeli alakulását.

**Jakab Gábor**

Devizaváltás

A modul feladata, hogy aktuális és történeti devizaárfolyamokat kezeljen, valamint lehetőséget adjon összegek átváltására különböző pénznemek között.

Az árfolyamokat egy külső API-ból olvassa be, és helyileg tárolja gyors eléréshez.

**Fő funkciók**

* Árfolyam-letöltés API-ból
* Konverzió

convert(amount, from\_ccy, to\_ccy, date=None) → float

történeti dátum esetén a legközelebbi elérhető nap árfolyama

Többszörös lánckonverzió kerülése: minden esetben bázisdevizán keresztül (pl. HUF↔EUR).

* Tesztelés

Unit tesztek: konverziós képletek, kerekítés, dátumkezelés.

Integrációs tesztek: mock API válaszok (aktuális és történeti).

Visszamenőleges pontosság ellenőrzése ismert mintákkal.