

„Szakkoli” csapat

Varga Krisztina (Project manager)

Takács Tamás

Szabó Bence

Rendszerterv

Portfólióválasztó alkalmazás

Tartalom

[Bevezetés 1](#_Toc468403580)

[Rendszerkövetelmény 1](#_Toc468403581)

[Alkalmazás architektúra 1](#_Toc468403582)

[Presentation Layer 2](#_Toc468403583)

[Business Layer 2](#_Toc468403584)

[Adatbázis réteg 4](#_Toc468403585)

[Összefoglalás: 5](#_Toc468403586)

# Bevezetés

A projekt célja egy backend php alapon működő, megtakarítási portfóliókat matematikailag összehasonlító alkalmazás fejlesztése. A megadott paramétereken alapuló döntést akarunk hozni, hogy a legjobb portfóliót alkalmazzuk, így maximalizálva a profitot. A véletlen hatásoktól eltekintünk. A program összeveti a diszkrét (jeled: d) illetve a folytonos (jele: f) idősorokat, így a következő bemenetekkel rendelkezik: kamatláb(d), kamatláb(f), időtartam(d),időtartam(f). Az időtartam(f) tulajdonképpen több időmértéket tartalmazhat attól függetlenül, hogy féléves, negyedéves stb. az idő, amikor a pénzünket kivesszük (kamatos kamat). Összehasonlítjuk az eredményeket, értékeljük azokat és kiszámítjuk folytonos idősornál azt a legkisebb időtartamot, ahol egyezés történik a diszkrét esettel. Emellett a kapott eredményeket egy egyszerű php oldalon megjelenítjük. A projekt során követelmény az analitikai eredmények mysql adatbázisban történő tárolása. A projekt nem terjed ki a publikus webszerveren történő elérésre, így a megvalósítás során localhost-on készítjük el az alkalmazást, amelyet a megrendelő a megadott követelmények teljesítése mellett migrálhat a saját környezetébe. Továbbá a mivel a megrendelés tárgya a matematikai analízis megvalósítása, így a felhasználói felületet nem valósítjuk meg a projekt során.

# Rendszerkövetelmény

- Apache 2 webszerver

- PHP 5.2.0 szervermodul

- MySQL adatbázis

- publikus domain név

- ajánlott HTML5 weboldal beágyazáshoz

- Tárhely a HTML5 méretének figyelembevételével.

# Alkalmazás architektúra

Az alkalmazás fejlesztéséhez a három rétegű architektúrát használjuk, felülről lefelé haladva a felső szint az megjelenítő réteg. Attól eltekintve, hogy a megrendelő nem kérte Presentation Layer fejlesztését, a későbbi könnyű továbbfejlesztés érdekében létrehozunk egy megjelenítő oldalt ahol megadhatók az elemzés paraméterei. Ez alatt helyezkedik el a Business Logic, amely az elemzéseket végzi. Végül legalul helyezkedik el a Data Access Layer, amely menti a számítási eredményeket a MySQL adatbázisba.

1. ábra: A háromrétegű architektúra modellje

# Presentation Layer

A presentációs réteghez a későbbi könnyű migrálhatóság érdekében index.html front controllert használunk, amelyen keresztül megadhatóak a matematikai analízis paraméterei. A továbbítása metódus „POST” az action metódus elérési útja pedig a main.php analízis script. Az adatbeviteli oldal input mezői a következők:

|  |  |  |  |
| --- | --- | --- | --- |
| **Megjelenített szöveg** | **type** | **name** | **value** |
| Havi megtakarítás: | int | megtakaritas |  |
| Diszkrét kamatláb: | int | d\_kamat |  |
| Folytonos kamatláb: | int | f\_kamat |  |
| Futamidő: | int | ido |  |
| Heti kamat | radio | check\_listd | P1W |
| Havi kamat | P1M |
| Negyedéves kamat | P3M |
| Féléves kamat | P6M |
| Éves kamat | P1Y |
| Heti kamat | checkbox | check\_listf[] | P1W |
| Havi kamat | P1M |
| Negyedéves kamat | P3M |
| Féléves kamat | P6M |
| Éves kamat | P1Y |
| Küldés | submit |  |  |

A megadott mezők megfelelő kitöltésével a Business Layer már hiba nélkül tud számítást végezni.

# Business Layer

Ebben a rétegben a script létrehoz egy portfolio objectumot, amely hordozza a beviteli oldalon megadott adatokat, és az objektum osztálymetódusaival számítja a kívánt eredményt.

A portfolio objektum lehetséges tagváltozói:

private $stock; - Havi megtakarítás

private $d\_interest; - diszkrét kamatláb

private $c\_interest; - folytonos kamatláb

private $duration; - futamidő

private $intervals; - választott folytonos kamatozási intervallum

private $begin; - aktuális rendszeridő

private $end; - aktuális rendszeridő + futamidő

A portfolio objektum példányosítása mellett érdemes inicializálni a PHP Dateinterval-okat, amelyek bizonyos időintervallumot tartalmaznak, a beviteli mezőben az ezekhez tartozó reguláris kifejezéseknek megfelelően deklaráljuk a visszaadott értékeket.

Ezután számítjuk a diszkrét idősorokat, amelyeknek a visszatérési értéke egy kétdimenziós tömb (igény esetén asszociatív) lesz amelynek egy dimenzióra eső elemei a kamatozási periódus aktuális időpontja, és a mindenkori időpont aktuális megtakarítási értéke lesz forintban.



2. ábra: diszkrét idejű kamatozás folyamatábrája

A diszkrét idejű kamatszámítás mellett számítjuk a folytonos idejű kamatozás forma idősorát is a 3. ábrán megadott folyamatábra szerint.



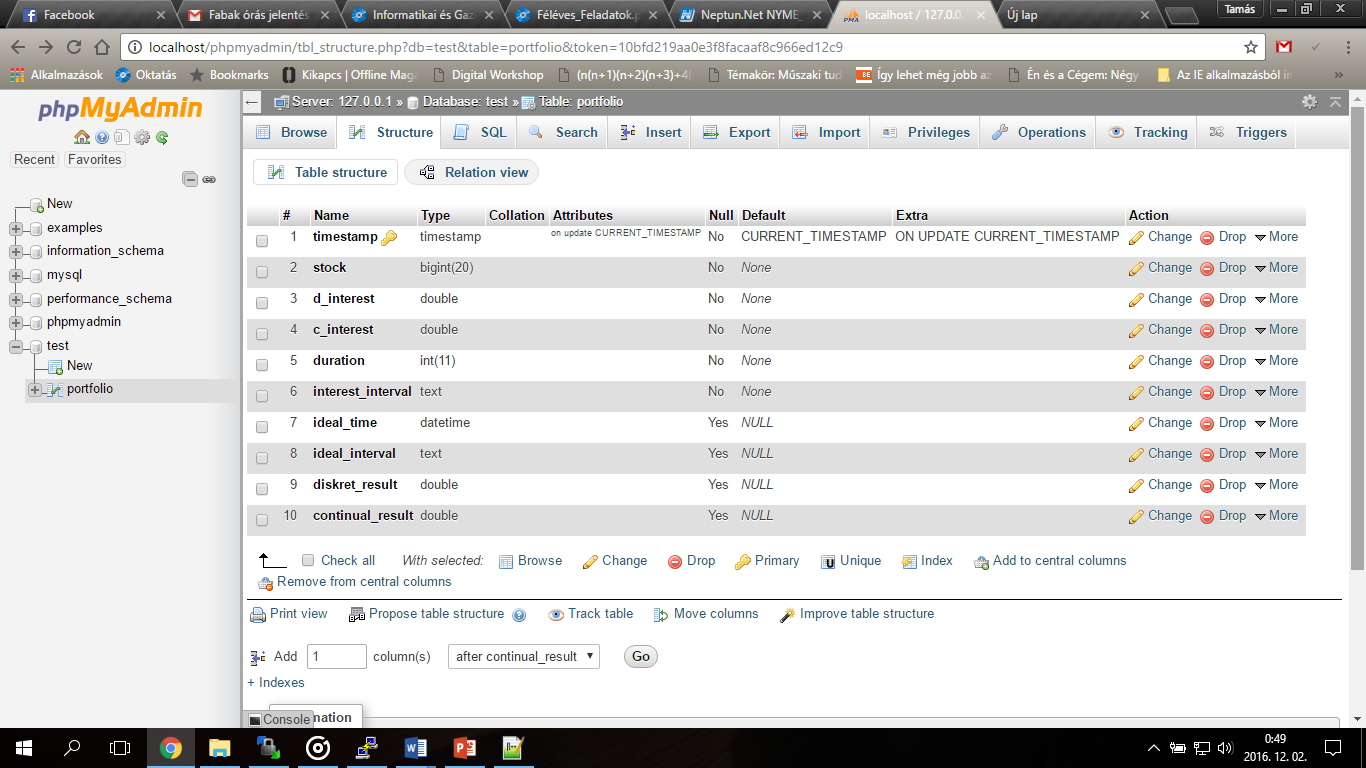
3. ábra: folytonos idejű kamatozás számítása

Amint előállt a diszkrét és a folytonos idősorokat tartalmazó tömb is, ezeket egy függvénnyel össze kell hasonlítanunk, és ha annál az $i eleme (folytonos idősor) elemnél, ahol azonos időpillanatban a folytonos megtakarítás nagyobb hozamot ér el, mint a diszkrét eset, adatbázisba kell írnunk az eredményt,és a return utasítással ki kell lépnünk függvényből.

# Adatbázis réteg

A követelményeknek megfelelően a matematikai analízis eredményeit MySQL adatbázisban kell tárolnunk. Mivel a PHP képes adatbázis tranzakciókat végezni, ezért ezeket beépített függvényekkel elvégezhetjük.

Az adatbázis egy táblát igényel, amely tárolja a beviteli adatokat és a eredményeket is. Az adattábla struktúra az 4. ábrán megtekinthető.



4. ábra: Adattábla struktúra

# Összefoglalás:

A megadott terv segítségével a megrendelő egyszerűen migrálhatja a kész alkalmazást a saját környezetében, amennyiben teljesítette a megadott rendszerkövetelményeket. Az elkészített default HTML form kibővítésével megfelelő felhasználói felületet adhat meg a Business Logic réteg fölé, a saját adattábla importálásával, és az authentikációs adatok személyre szabásával futtathatóvá tehető az alkalmazás.

**Készítette:**

**Takács Tamás**

**Varga Krisztina**

**Szabó Bence**