**EÖTVÖS LORÁND TUDOMÁNYEGYETEM**

Informatika Kar

Programozáselmélet és Szoftvertechnológia Tanszék

MultiRetro – Agilis retrospektívek levezetését támogató eszköz

**Témavezető:** **Szerző:**

Ilyés Enikő Balogh Emese

PhD hallgató Programtervező informatikus BSc.

**Budapest, 2021**

**Témabejelentő**

A MultiRetro egy olyan webes eszköz, amely támogatja az agilis csapatokat változatos retrospektív technikák alkalmazásában és lehetőséget nyújt akciópontok létrehozására a retrospektívek nyomán.

A honlapon elérhető a regisztráció és a belépés funkciója, hiszen a MultiRetro funkciói regisztrált felhasználók számára biztosítottak. A regisztrált személyekből különböző csapatok formálására van lehetőség. A csapatok számára retrospektív események hozhatók létre, változatos retrospektív sémák generálásával. Ezek a sémák segítik hatékony retrospektívek levezetését. A retrospektívek során bejött adatok szóbeli elemzésével párhuzamosan a résztvevők akciópontokat hozhatnak létre a folyamataik optimalizálását illetően. Ezeket egy Kanban táblában jeleníti meg a rendszer és a csapattagoknak lehetőségük van a retrospektívek között rápillantani ezekre az akciópontokra és állapotaikra (“Megvalósításra vár”, “Megvalósítása folyamatban”, “Megvalósítva”), valamint módosítani állapotaikat.

A fő funkciókon kívül biztosítva van retrospektívek témakörét érintő, informatív anyagok blogbejegyzés formájú írása és megtekintése. A blogbejegyzések között a felhasználók különböző címkék alapján tudnak keresni. A csapatok támogatását illetően a naplózás funkció is segítségül szolgál. A honlap biztosítja a karbantartóival való kapcsolatfelvétel lehetőségét is.

A MultiRetro Laravel és egyéb webes technológiák segítségével van megvalósítva, és agilis csapat által tesztelve.

Tartalomjegyzék

[Köszönetnyilvánítás 3](#_Toc71632605)

[1. Bevezetés 4](#_Toc71632606)

[2. Felhasználói dokumentáció 5](#_Toc71632607)

[2.1 Rendszerkövetelmények 5](#_Toc71632608)

[2.2 Az alkalmazás használata 6](#_Toc71632609)

[2.2.1 Kezdőlap 6](#_Toc71632610)

[2.2.2 Bejelentkezés és regisztráció 6](#_Toc71632611)

[2.2.3 Profil 7](#_Toc71632612)

[2.2.4 Csapatok 8](#_Toc71632613)

[2.2.5 Megbeszélések 9](#_Toc71632614)

[2.2.5 Akciópontok 16](#_Toc71632615)

[2.2.5 Blogok 18](#_Toc71632616)

[2.2.5 Napló 19](#_Toc71632617)

[3. Fejlesztői dokumentáció 21](#_Toc71632618)

[3.1 Fejlesztő környezet 21](#_Toc71632619)

[3.2 Drótváztervek 21](#_Toc71632620)

[3.3 Grafika 25](#_Toc71632621)

[3.3.1 Kezdőlapon szereplő kép – cat\_puzzle.png 25](#_Toc71632622)

[3.3.2 Időjárás-jelentés ikonok 26](#_Toc71632623)

[3.4 Használati esetek 27](#_Toc71632624)

[3.5 Architektúra 29](#_Toc71632625)

[3.6 Adatbázis 30](#_Toc71632626)

[3.6.1 Az adatbázis felépítése 30](#_Toc71632627)

[3.6.2 Egyed-kapcsolat diagram 31](#_Toc71632628)

[3.6.3 Migration és Model 32](#_Toc71632629)

[3.6.4 Seeding 35](#_Toc71632630)

[3.7 Controllers and routes – kontrollerek és útvonalak 36](#_Toc71632631)

[3.7.1 UserProfileController 36](#_Toc71632632)

[3.7.2 TeamController 37](#_Toc71632633)

[3.7.3 MeetingController 38](#_Toc71632634)

[3.7.4 ActionpointController 40](#_Toc71632635)

[3.7.5 DiaryController 41](#_Toc71632636)

[3.7.6 PlusMinusTaskController 41](#_Toc71632637)

[3.7.7 BlogController 42](#_Toc71632638)

[3.8 Views - nézetek 43](#_Toc71632639)

[3.9 Segédfüggvények 46](#_Toc71632640)

[3.10 Bővítési lehetőségek 48](#_Toc71632641)

[3.11 Tesztelés 49](#_Toc71632642)

[3.11.1 Automatikus tesztek 49](#_Toc71632643)

[3.11.2 Manuális tesztek 52](#_Toc71632644)

[4. Forrásjegyzet 59](#_Toc71632645)

# Köszönetnyilvánítás

Köszönöm szépen, hogy BSc-s hallgatóként lehetőséget kaptam arra, hogy az Agilis kutatócsoport tagja lehessek. Hálás vagyok, hogy részt vehettem a heti megbeszéléseken, és a többi hallgatótársam megengedte, hogy belelássak munkafolyamataikba. Nagyon sok új dolgot tanultam Tőletek, és rengeteg hasznos tanáccsal láttatok el.

Külön köszönet a tesztvideókban való részvételért:

* Ilyés Enikő
* Bornemissza Barbara
* Cseh-Szabó Lilla
* Geri Botond Máté
* Gulyás Gergő
* Tóth Péter

# 1. Bevezetés

A szakdolgozatban bemutatott MultiRetro webes alkalmazás az agilis retrospektívek levezetését támogató eszköz.

Az alábbiakban tisztázom a legfontosabb fogalmakat:

* Agilis módszertan: gyors és hatékony fejlesztést tesz lehetővé az alábbi négy tényezőt hangsúlyozva: az egyéneket és a személyes kommunikációt a módszertanokkal és eszközökkel szemben; a működő szoftvert az átfogó dokumentációval szemben; a megrendelővel történő együttműködést a szerződéses egyeztetéssel szemben; a változás iránti készséget a tervek szolgai követésével szemben.[[1]](#footnote-1)
* Retrospektív: visszatekintés, az agilis módszertanoknál jellemző esemény, ahol a fejlesztő csapat egy visszatekintő megbeszélést tart egy elmúlt munkaszakaszra vonatkozóan. Visszatekintenek a munkaszakasz során felmerült problémákra és javaslatot tesznek munkaszervezésük javítására vonatkozóan. A Scrum módszertan esetén minden futam (előre rögzített, azonos hosszúságú munkaszakasz) végén Scrum Retrospektívet tart a csapat.[[2]](#footnote-2)

Az egyetemi tanulmányaim 4. félévében találkoztam először olyan csapatmunkával, ahol a szükséges irányítást biztosító módszertan nélkül nem tudtuk volna időben elvégezni a feladatunkat. Scrum elemeket tartalmazó agilis módszertant használtunk. Tapasztalatom szerint, ahhoz, hogy a közös munka hatékony legyen, fel kell állítani célokat, terveket, ugyanakkor az egyik legfontosabb, hogy egy munkafolyamat végén osszuk meg egymással a tapasztalatainkat, beszéljük át a felmerülő problémákat, és keressünk rájuk megoldást.

Az általam szakdolgozatként megvalósított webes alkalmazás segítségével egy olyan felületet biztosítok a felhasználóknak, ahol lehetőségük van különféle retrospektív eszközök használatára. Ezen kívül a retrospektív megbeszélés során akciópontokat is létrehozhatnak a résztvevők, melyek később visszanézhetők és módosíthatók.

Regisztráció után van lehetőség a csapatok formálására, megbeszélések létrehozására, blogbejegyzések írására, valamint a naplózás fül alatt meg lehet tekinteni a lezárt megbeszélések naplózott adatait.

Összegezve, a MultiRetro egy olyan felületet biztosít csapatok számára, mely segít hatékonyabbá tenni a közös munkájukat a különböző funkciók segítségével.

# 2. Felhasználói dokumentáció

## 2.1 Rendszerkövetelmények

A MultiRetro alkalmazást a következő oldalon lehet elérni: <http://multiretro.herokuapp.com/public/>

Az alább leírt teszt felhasználó segítségével az alkalmazás kipróbálható:

e-mail cím: admin@admin.hu, jelszó: adminadmin

A MultiRetro tesztelése elsősorban Google Chrome-mal zajlott, így ez az elsődlegesen támogatott böngésző. Reszponzív felületet biztosít a webes alkalmazás, így számítógépen, tableten és mobiltelefonon egyaránt kényelmes a használata.

Ha a felhasználó a forráskód alapján szeretné futtatni a programot lokális környezetben, akkor erre is van lehetőség [3]. Az alkalmazás kódja GitHub-on elérhető a következő linken: <https://github.com/baloghemi/szakdoga_repo>

Ha a környezeti változók megfelelőek, akkor nyissuk meg a terminált és lépjünk be a multiretro mappába. Itt az alábbi parancsokat szükséges beírni:

1. composer install
2. npm install
3. php artisan migrate
4. php artisan db:seed
5. php artisan serve

Az utolsó parancs után a weblap elérhető lesz a localhost:8000 vagy <http://127.0.0.1:8000> porton.

Ha a kódot GitHub-ról töltjük le, akkor az előbb említett lépések mellett, pontosan a 2. lépés után szükséges az env fájl létrehozása és a kulcs generálása:

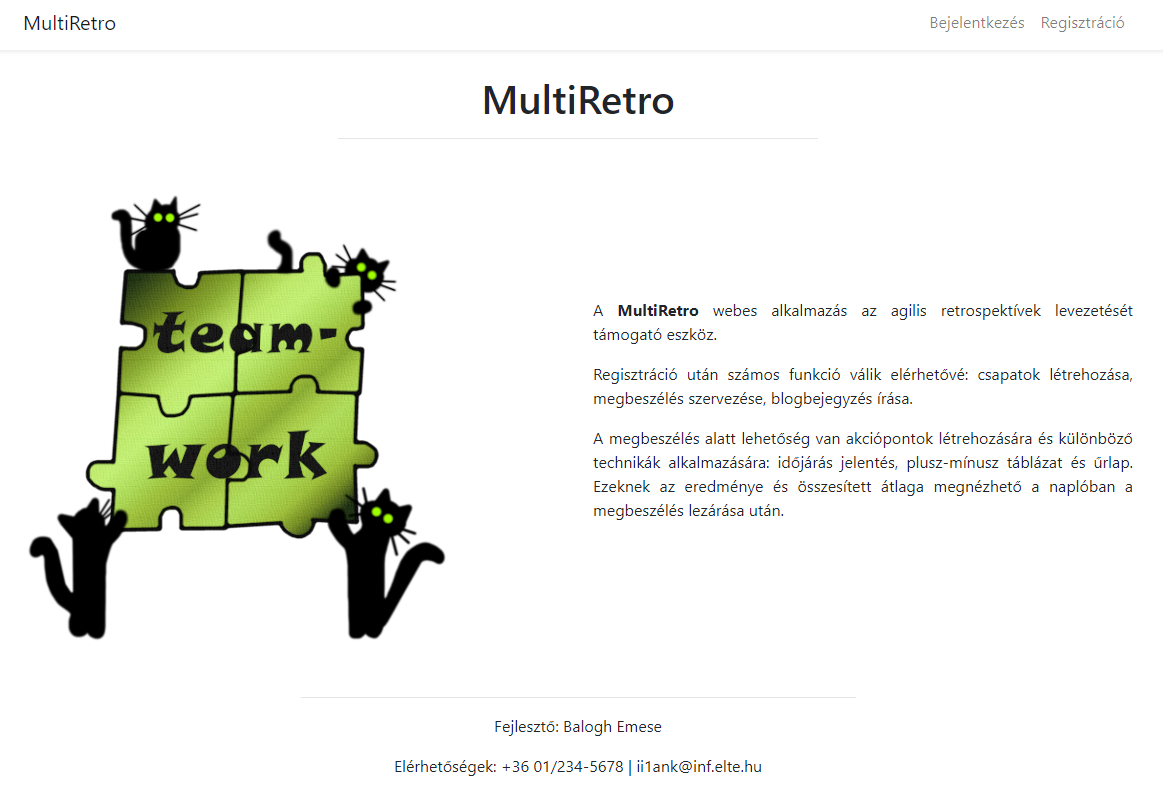
* cp .env.example .env
* php artisan key:generate

A telepítéssel kapcsolatban további információkat lehet találni a Laravel dokumentációban a Getting started menüponton belül az Installation és Deployment leírásokban.[[3]](#footnote-3)

## 2.2 Az alkalmazás használata

### 2.2.1 Kezdőlap

A kezdőlapon található egy kép és egy rövid leírás a weblap főbb funkcióiról. Ahogy az első ábrán látható, a felső jobb sarokban van a Bejelentkezés és a Regisztráció opció. Ezek akciógombok, így rájuk kattintva tovább viszik a felhasználót a megfelelő oldalra. Bejelentkezés után elérhetővé válik az összes navigáció: Profil, Csapatok, Megbeszélések, Akciópontok, Blogok, Napló és Kijelentkezés. A MultiRetro feliratra kattintva mindig a kezdőlapra jutunk.



1. ábra: Kezdőlap nézet

### 2.2.2 Bejelentkezés és regisztráció

Bejelentkezés (lásd: 2. ábra):

Kettő bemeneti mező jelenik meg, az e-mail cím és a jelszó. Ha a felhasználó el van mentve az adatbázisban, akkor probléma nélkül bejelentkezik. Ha esetleg olyan adatokat ad meg, amik még nincsenek regisztrálva vagy rosszul gépelte be valamelyik adatot, akkor hibaüzenetet kap.

Regisztráció (lásd: 3. ábra):

A regisztrációhoz négy darab mezőt kell kitölteni: Név, e-mail cím, jelszó és jelszó megerősítése. Ha valamelyik bemenet nem felel meg a követelményeknek, akkor hibaüzenet kap a felhasználó. Sikeres bejelentkezés vagy regisztráció után a Profil oldal jelenik meg.

|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\bejelentkezés_telefon.png  2. ábra: Bejelentkezés - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\regisztráció_telefon.png  3. ábra: Regisztráció - mobil nézet |

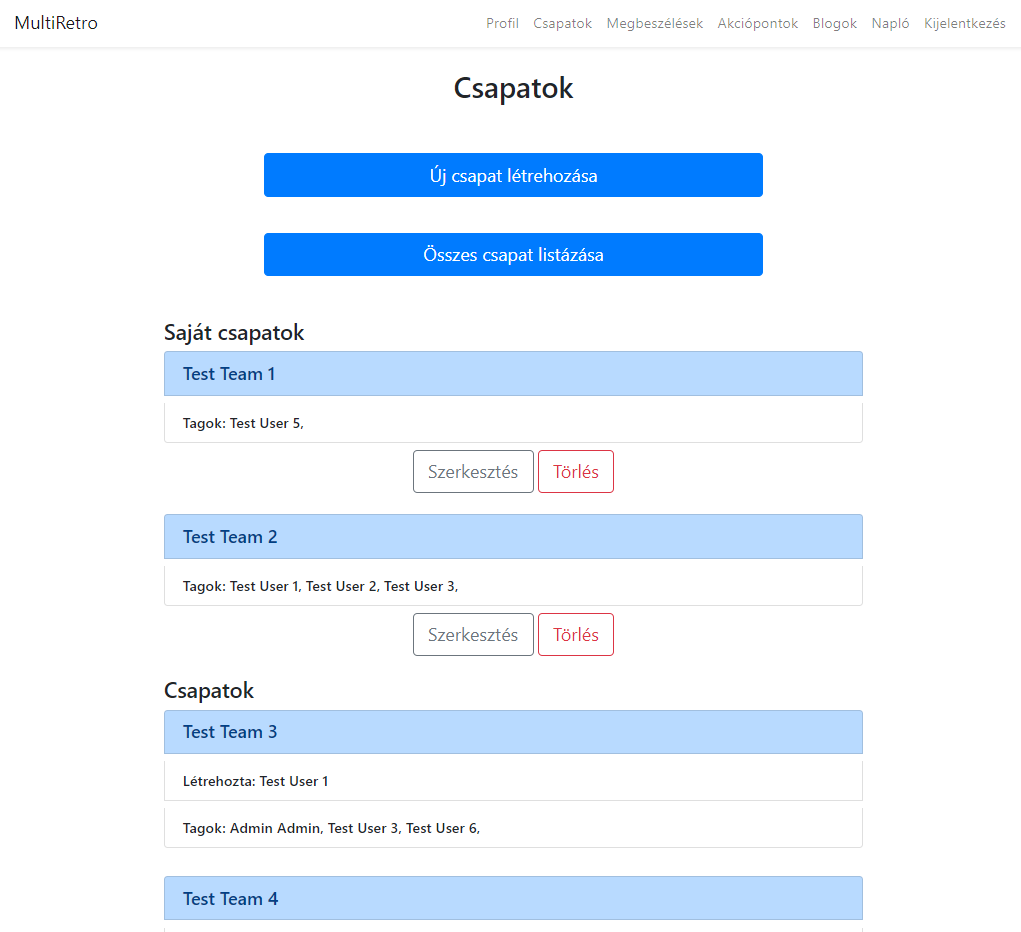
### 2.2.3 Profil

Ahogy a 4-es ábrán látható, a ’Felhasználó adatai’ cím alatt a következő adatok sorakoznak fel: név, e-mail cím. A felsorolás alatt egy akciógomb van, melyre rákattintva megjelenik az 5-ös ábrán látható nézet, ahol módosítani lehet a felhasználó adatait a jelszó kivételével. Az ’Adatok módosítása’ vagy a ’Vissza’ gombra kattintva visszakerülünk a profil nézetre.

|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profil_mobil.png  4. ábra: Profil - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\profil_módosítás_mobil.png  5. ábra: Profil módosítása - mobil nézet |

### 2.2.4 Csapatok

A csapatok fülre kattintva kilistázódnak azok a csapatok, amiket a felhasználó hozott létre, valamint amelyeknek a tagjai között szerepel. Minden csapatot, amit a felhasználó hozott létre, szerkeszteni vagy törölni tudja a csapat alatt lévő megfelelő gombra kattintva. A lap tetején további két akciógomb látható: új csapat létrehozása és az összes csapat listázása. Mindez a 6-os ábrán található.



6. ábra: Csapatok nézet

Ahogy a 7-es ábra mutatja, új csapat létrehozásánál szükséges megadni a csapat nevét és a tagokat. Szerkesztésnél szintén ez a nézet jelenik meg, azonban az űrlap ki van töltve az adatbázisban szereplő adatokkal, így segítve a felhasználót a módosításnál. Az összes csapat listázásánál a csapat neve, létrehozója és tagjai jelennek meg (lásd: 8. ábra).

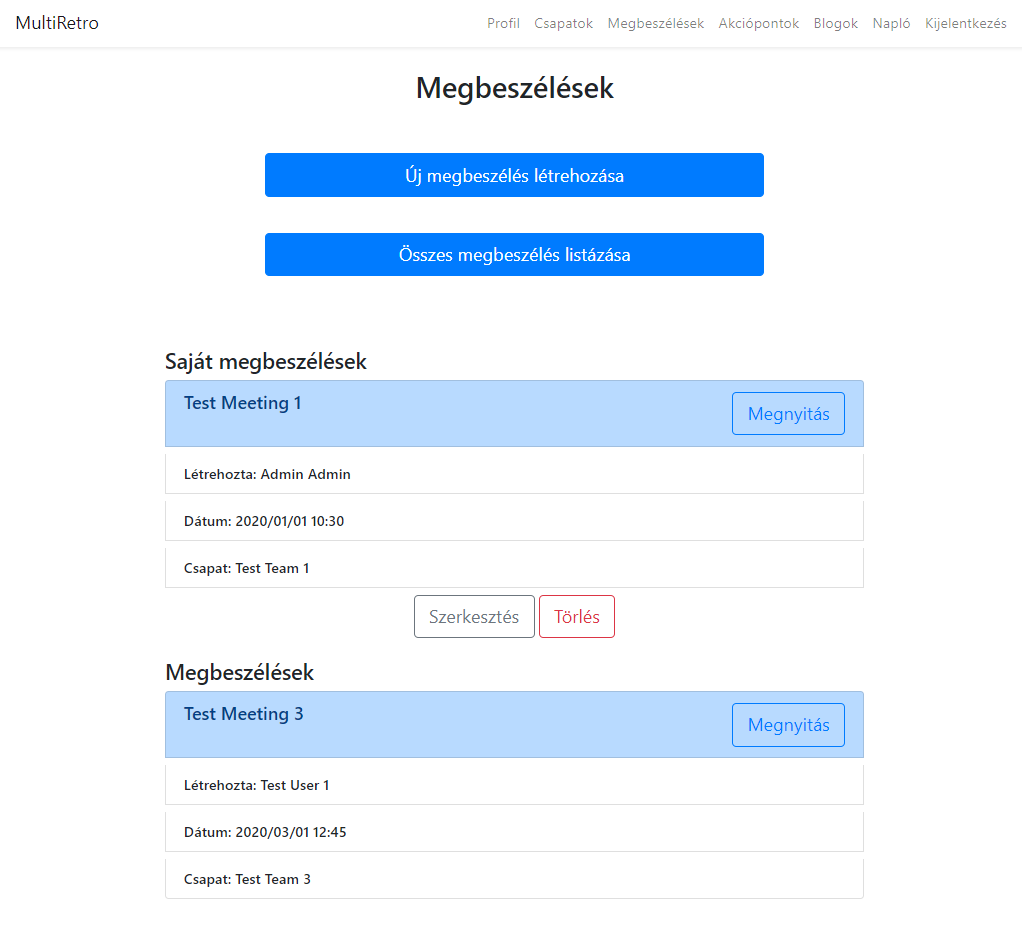
|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\csapatok_módosítás.png  7. ábra: Csapat módosítása - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\csapatok_lista.png  8. ábra: Csapatok listázása - mobil nézet |

### 2.2.5 Megbeszélések

A megbeszélés tervezését segítette Esther Derby és Diana Larsen könyve [7], valamint a scrum-tips.com oldalon található akciópontokról írt cikk [6].

A megbeszélés lapon - 9. ábra - listába rendezve találhatók a felhasználó által létrehozott megbeszélések, valamint azok, amelyeknek a tagja. Minden saját megbeszélés alatt van egy szerkesztés és egy törlés gomb, valamint a megbeszélés neve mellett egy megnyitás gomb.

A lista előtt található két akciógomb, a felső felel az új megbeszélés létrehozásáért, az alsó az összes megbeszélés megjelenítéséért, mely a 11-es ábrán látható. Minden megbeszélésnél látható a megbeszélés neve, létrehozója, a dátum és annak a csapatnak a neve, amelyhez a megbeszélés tartozik.



9. ábra: Megbeszélések nézet

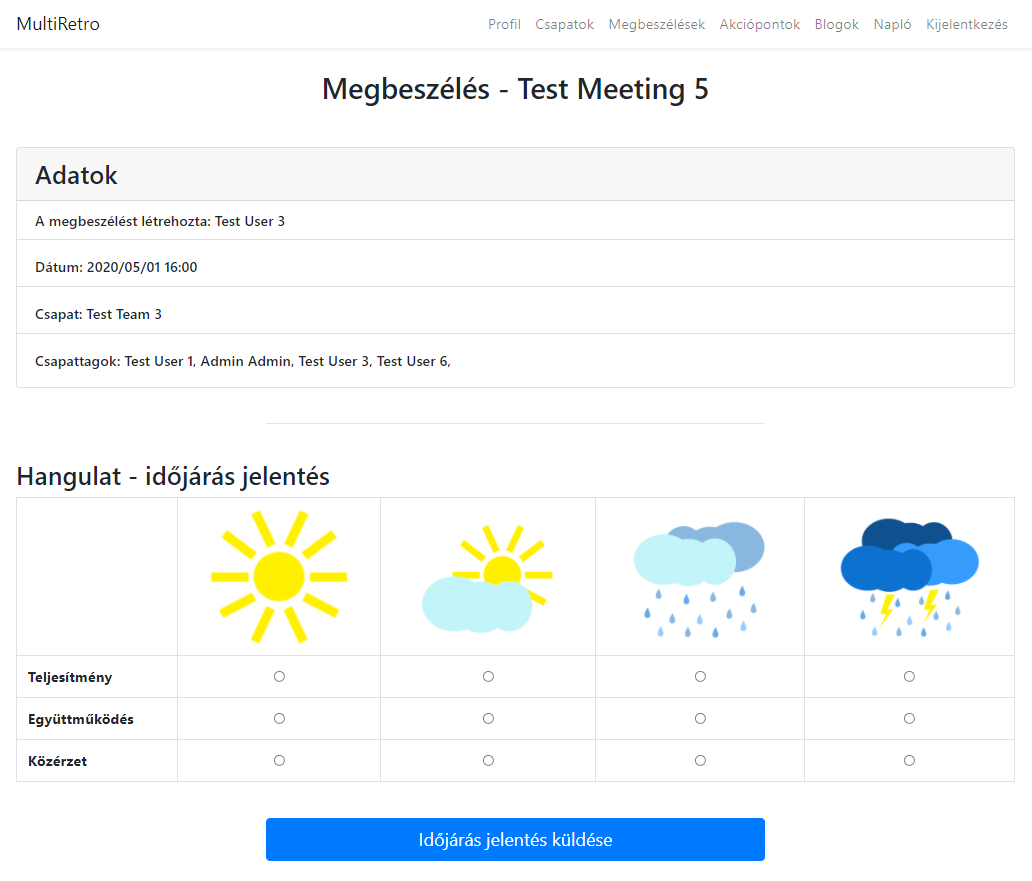
A szerkesztésért és az új megbeszélés létrehozásáért felelős nézet a 10. ábrán látható. Új megbeszélés létrehozásánál szükséges megadni a megbeszélés nevét, időpontját és a csapatot. Egy felhasználó több megbeszélést is létrehozhat, de minden megbeszéléshez csak egy csapat tartozhat. Szerkesztésnél a mezők ki vannak töltve a megfelelő adatokkal; ez segítségül szolgál a felhasználónak az adatok módosításában.

|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\megbeszélés_módosítás_mobil.png  10. ábra: Megbeszélés módosítása - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\megbeszélés_lista_mobil.png  11. ábra: Megbeszélések listázása - mobil nézet |

Minden megbeszélés mellett elérhető a megnyitás gomb, ami átirányítja a felhasználót a retrospektív levezetés oldalára. Itt több funkció is elérhető az adott csapat számára:

* adatok megjelenítése: megbeszélés létrehozója, dátum, csapat neve, csapattagok
* hangulat visszajelzése az időjárás-jelentés technikával
* pozitív és negatív élmények megosztása kártyák írása technikával
* több szempontú helyzetfelmérés űrlap kitöltése technikával
* Kanban tábla

A lap legalján található a ’Megbeszélés lezárása’ gomb, mely csak a megbeszélés létrehozójánál jelenik meg. A lezárt megbeszélés és a hozzá tartozó technikák által begyűjtött adatok csak a naplóban tekinthetők meg, nem lehet többé szerkeszteni őket.



12. ábra: Retrospektív felület – megbeszélés adatai és időjárás jelentés

**Időjárás-jelentés**

Az ún. időjárás-jelentés egy retrospektív technika – 12-es ábra – egyszerű rajzok használatával segít a csapattagok hangulatát kifejezni az adott csapatmunkával kapcsolatos területet illetően. Az ’Időjárás-jelentés küldése’ gombra kattintva elmentődnek a naplóba a felhasználó által megadott értékek, majd a gomb helyén megjelenik kettő átlag: a felhasználó átlaga és a csapat együttes átlaga.

Értékelésre váró területek:

* teljesítmény
* együttműködés
* közérzet

Az időjárás rajzok értelmezése következőképpen történik:

* napos: határozottan elégedett
* felhős: elégedett, de van, amin lehetne javítani
* esős: nem elégedett, de nem is elégedetlen, át kell beszélni pár dolgot
* viharos: elégedetlen, nagyon sok mindenen kell javítani

|  |  |  |  |
| --- | --- | --- | --- |
| Napos | Felhős | Esős | Viharos |
| C:\Users\Diak\AppData\Local\Microsoft\Windows\INetCache\Content.Word\sun.png |  |  | C:\Users\Diak\AppData\Local\Microsoft\Windows\INetCache\Content.Word\stormy.png |

A hangulat visszajelzést érdemes a táblázat kitöltése után szóban is átbeszélni. A következő kérdéseket érdemes feltenni:

1. Mi(k) a probléma oka(i)?
2. Mi a kívánt kimenetel?
3. Min változtatnál?
4. Hogyan tudjuk elérni a sikert?

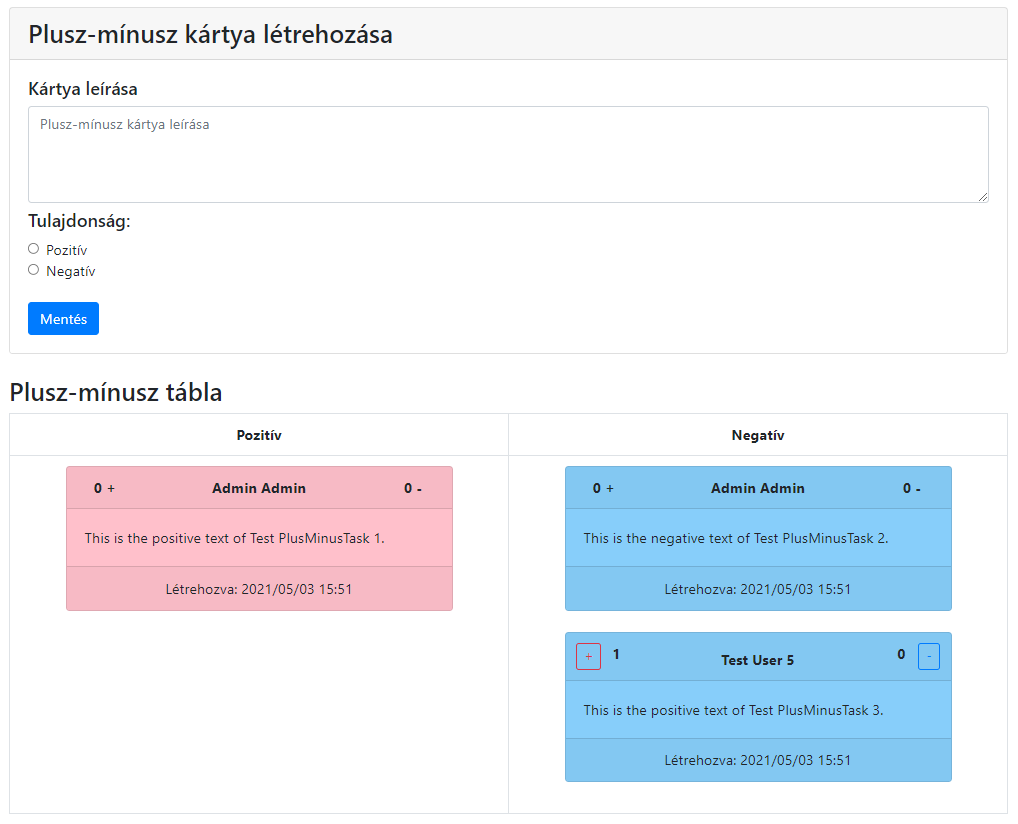
A hangulat visszajelzés lényege, hogyha esetleg probléma adódott valamelyik említett területen, akkor arra közösen megoldást találjon a csapat.

**Plusz-mínusz táblázat**

A plusz-mínusz táblázat két oszlopból áll, az egyik a pozitív oldal, a másik a negatív. Ahogy a 13-as ábrán látható, a táblázat felett lehetőség van új kártya létrehozására. A kártya leírását és a tulajdonságát szükséges ehhez megadni. Kezdetben minden kártyán 0 db plusz és 0 db mínusz jel található. A megbeszélésen résztvevő felhasználók szavazhatnak a kártyákra, hogy egyetértenek-e vele vagy sem, oly módon, hogy a kártyán rákattintanak a plusz vagy mínusz jelre. A felhasználó nem tud szavazni a saját maga által létrehozott kártyára.

A plusz-mínusz kártyák létrehozása előtt érdemes a csapatnak egy rövid felkészülési időt hagyni, hogy gondolkodhassanak azon, milyen fontosabb dolgokat szeretnének megosztani a többiekkel. Miután mindenki létrehozta azokat a kártyákat, amiket szeretne, megkezdődhet a plusz-mínusz értékelés.

A technika lényege, hogy a kiértékelés végére egy képet lehessen alkotni a pozitív és negatív elemekről, valamint az időjárás-jelentéshez hasonlóan, ha komolyabb probléma merül fel, akkor közösen megoldást találjon rá a csapat.



13. ábra: Retrospektív felület - plusz-mínusz kártyák

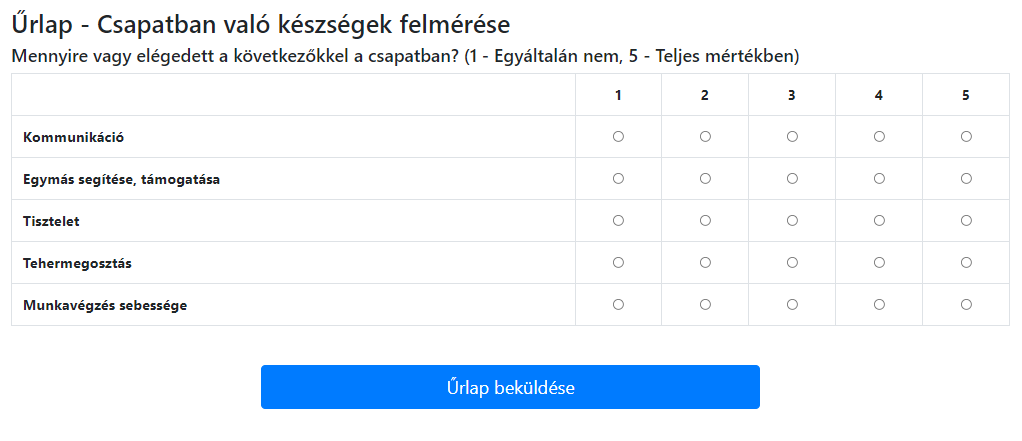
**Űrlap**

A harmadik választható technika egy egyszerű űrlap, mely a csapatban való készségeket méri fel. A készségek a következők:

* Kommunikáció
* Egymás segítése, támogatása
* Tisztelet
* Tehermegosztás
* Munkavégzés sebessége

Ezeket egy és öt között van lehetőség értékelni, hogy mennyire elégedett vele a csapattag. Az egyes érték jelenti az „Egyáltalán nem”-et, míg az 5-ös érték a „Teljes mértékben”-t.

Az ’Űrlap beküldése’ gombra kattintva a naplóba lementődnek a megfelelő adatok, és a táblázat helyén az „Az űrlap sikeresen el lett mentve!” felirat jelenik meg, valamint egy összesítő eredmény felület a felhasználó saját, a csapat együttes és a készségek szerinti átlagáról.

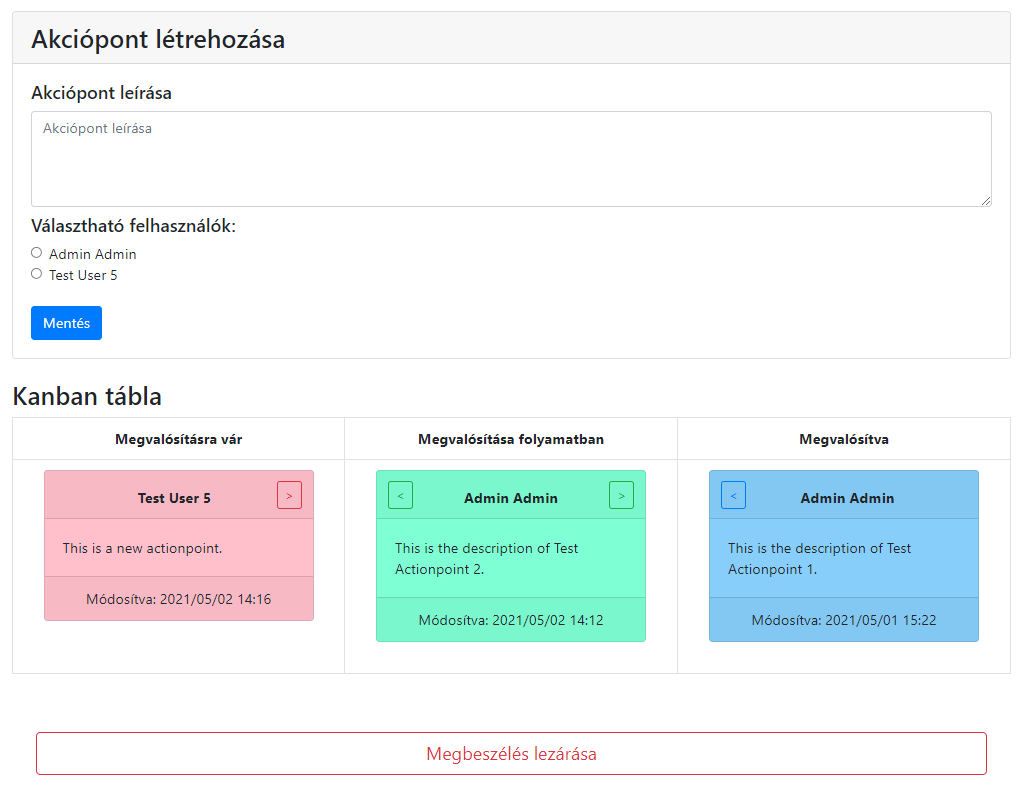


14. ábra: Retrospektív felület – Űrlap

**Kanban tábla és akciópontok**

A többi retrospektív technikával ellentétben, a Kanban tábla és a hozzá tartozó akciópontok létrehozása nem választható technika, minden megbeszélés megnyitásakor megjelenik. Ahogy a 15-ös ábrán látható, először az akciópont létrehozás, majd a Kanban tábla jelenik meg a következő fejlécekkel: „Megvalósításra vár”, „Megvalósítása folyamatban”, „Megvalósítva”.

Akciópont létrehozásnál szükséges megadni annak leírását, és a célszemélyt, akinek a feladatot szánja a csapat. A státusz alapértéke mindig „Megvalósításra vár”, így létrehozás után rögtön az első oszlopba fog kerülni a kártya. Természetesen lehetőség van ennek módosítására a kártyákon lévő nyilak használatával. A jobb nyíl egy oszloppal jobbra helyezi át a kártyát, míg a bal nyíl egy oszloppal balra.



15. ábra: Retrospektív felület - Kanban tábla és akciópontok

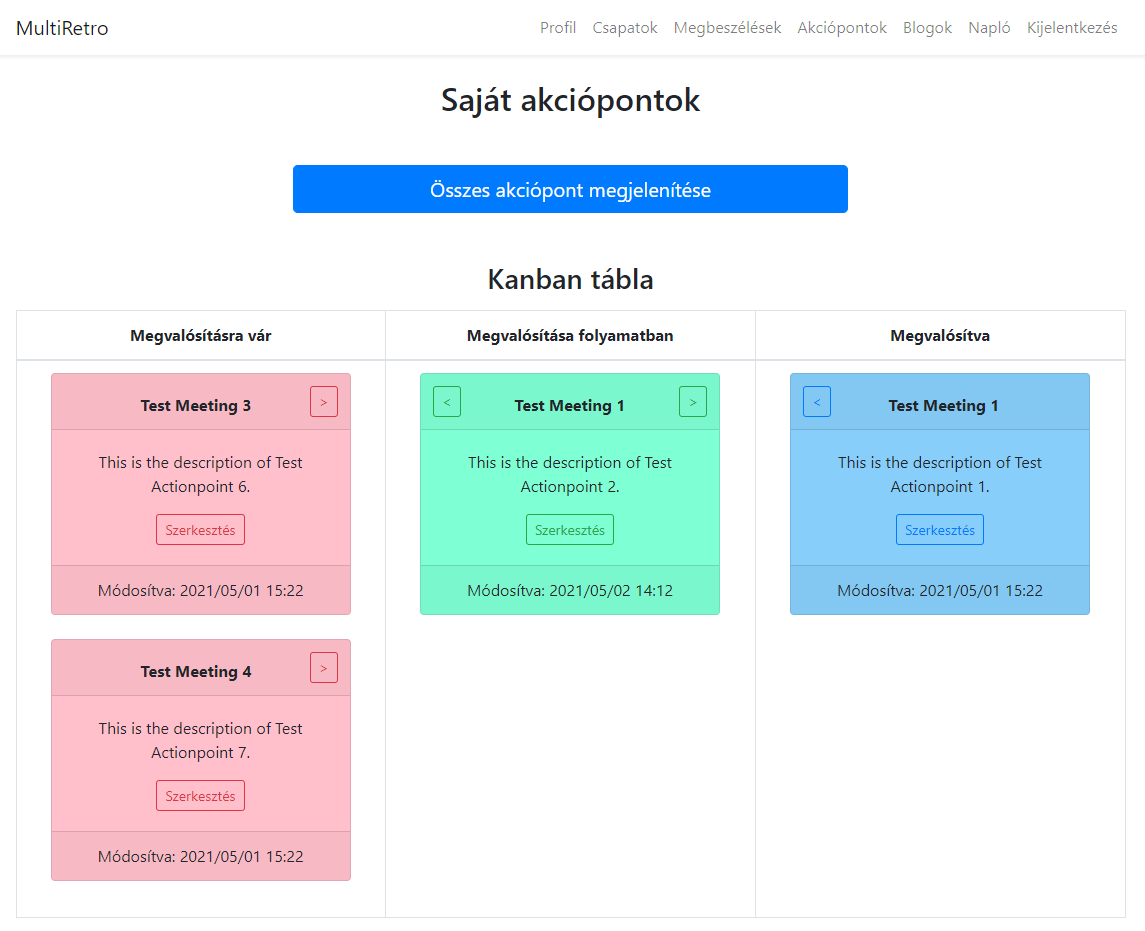
A technika lényege, hogy az akciópontok segítségével a csapat rendszerezni tudja a munkaszervezés javítására vonatkozó feladatokat, és kész terveket hozzanak létre a csapattagokhoz párosítva. Akciópontot csak megbeszélés során lehet létrehozni, viszont az Akciópontok lapon lehetőség van szerkeszteni őket.

**Megbeszélés lezárása**

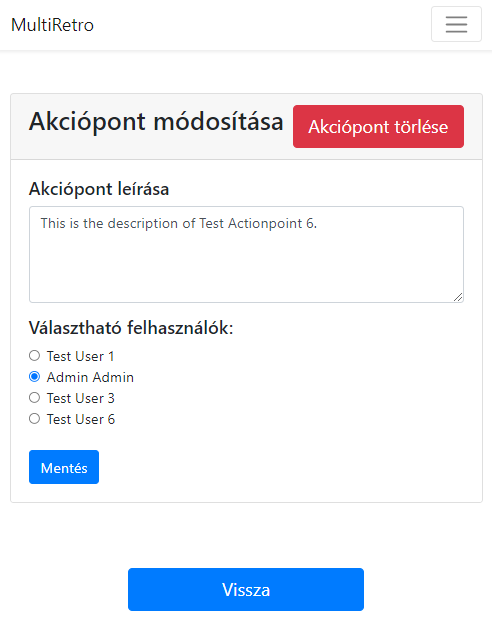
Csak az a felhasználó zárhatja le a megbeszélést, aki létrehozta. Lezárás után az adott megbeszélés adatai nem szerkeszthetők többé, csak a naplóban tekinthetők meg. Se a Megbeszélés, se az Akciópont lapon belül nem lesznek láthatók a megbeszéléshez tartozó adatok.

### 2.2.5 Akciópontok

Az Akciópontok lapon egy Kanban táblába rendezve találhatók meg a felhasználóhoz tartozó akciópontok. Ahogy a 16-os ábrán látható, hogy a kártyák fejlécén található a megbeszélés neve - ahol létre lett hozva, alatta a leírása, egy szerkesztés gomb és legalul az utolsó módosítás dátuma.



16. ábra: Akciópontok nézet

Ahogy a retrospektív felületnél volt tapasztalható, itt is biztosítva van a lehetőség, hogy a kártyák tetején lévő nyilakra kattintva mozgassuk az akciópontokat a tábla oszlopai között.

Új akciópontot itt nem lehet létrehozni, ez csak a megbeszélés keretein belül valósítható meg, azonban szerkesztésre van lehetőség a ’Szerkesztés’ gombra kattintva. Lehet módosítani az akciópont leírását és a hozzá tartozó felhasználót, valamint ezen a felületen lehet törölni az adott akciópontot. Mindez a 17-es ábrán figyelhető meg.

17. ábra: Akciópont módosítása - mobil nézet

### 2.2.5 Blogok

Ahogy a 18-as ábrán látható, a Blogok lapon találhatók meg a felhasználók által írt blogbejegyzések. A lap tetején két akciógomb van, a felső felel az új blog létrehozásáért, míg az alsó a felhasználó által írt blogok megjelenéséért. Az akciógombok alatt van egy kereső felület, ahova a keresőszó beírása után csak azok a blogok listázódnak ki, amelyek elmentett címkéi között ez a szó megtalálható. Minden bloghoz három darab címkét lehet lementeni, azonban csak egy darab keresőszó megadása kötelező, a másik kettő kitöltése választható. Továbbá új blog létrehozásakor szükséges megadni a címet és a leírást, ahogy a 19-es ábrán megfigyelhető. Ha egy már meglévő blogot szerkeszt a felhasználó, akkor a mezők automatikusan kitöltődnek a megfelelő adatokkal.

|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\blogok_mobil.png  18. ábra: Blogok - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\blog_módosítás_mobil.png  19. ábra: Blog módosítás - mobil nézet |

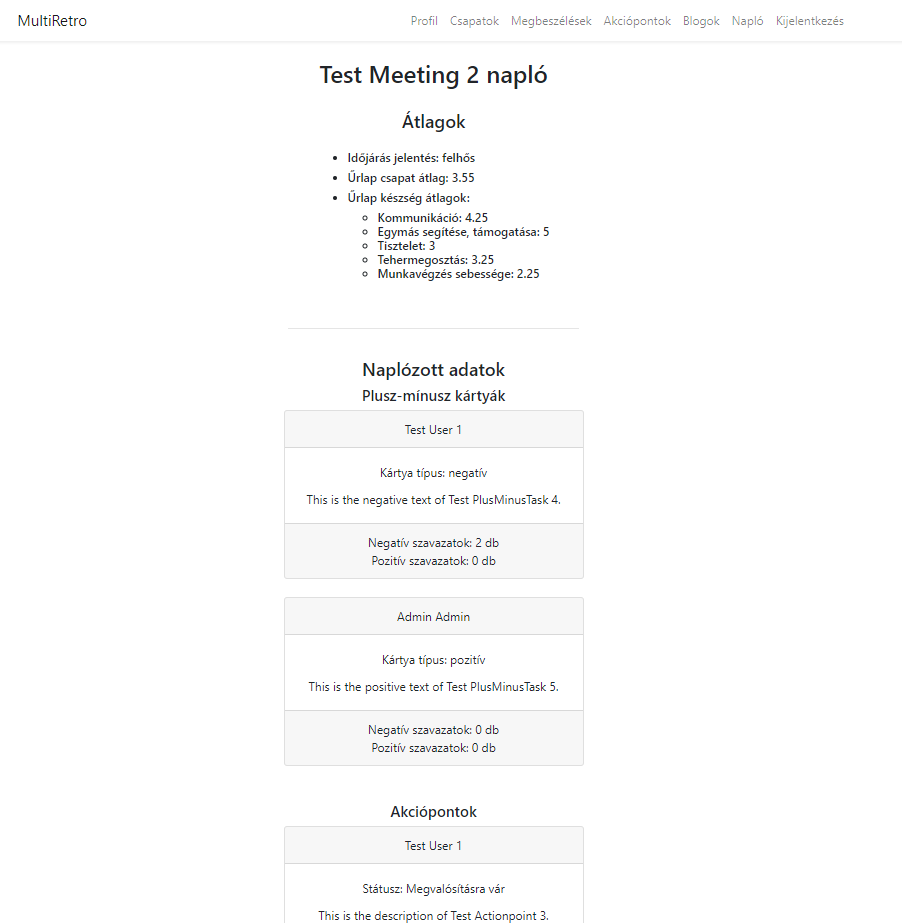
A saját blogok kilistázásánál (lásd: 20. ábra) megjelenik minden blogbejegyzés, amelyet a felhasználó hozott létre. Ezek rendezve vannak időrendi sorrendbe, mindig a legfrissebb, azaz a legutóbb módosított blog kerül legfelülre. Minden blog alatt található két gomb, az első felel a szerkesztésért, míg a második a törlésért.

|  |  |
| --- | --- |
| C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\blogok_saját_mobil.png  20. ábra: Saját blogok - mobil nézet | C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\napló_mobil.png  21. ábra: Napló - mobil nézet |

### 2.2.5 Napló

A Napló oldal felel a lezárt megbeszélések adatainak megtekintéséért. Ahogy a 21-es ábrán látható, a lap tetején vannak kilistázva azok a megbeszélések, amik le vannak zárva, míg alatta a felhasználó saját naplózott adatai tekintők meg. Itt megbeszélésenként vannak csoportosítva a technikákhoz tartozó eredmények.

A megbeszélés nevére kattintva jelenik meg a hozzá tartozó csapat lementett értékei. Először a különböző átlagok jelennek meg: időjárás jelentés, űrlap csapat és űrlap készség átlagok. Ezek alatt listába rendezve láthatók a 22-es ábrán megjelenített módon a plusz-mínusz kártyák és az akciópontok. Itt egyik érték sem módosítható, csupán megtekinthető.



22. ábra: Megbeszélés napló nézet

A felhasználók egyéni véleményének megvédése érdekében a különböző technikákra adott egyéni válaszok – időjárás jelentés, űrlap - nem nézhetők vissza, csak az átlagban mért értékük tekinthető meg. Egyedül az adott felhasználó tudja megnézni a naplóban, hogy ő személyesen milyen értékeket, hozzászólásokat adott meg.

# 3. Fejlesztői dokumentáció

A következőkben az alkalmazás fejlesztői szempontú felépítését mutatom be.

A fejezetben található megértést segítő ábrákat és diagramokat Draw.io[[4]](#footnote-4) segítségével készítettem el, kivéve ahol más forrást jelölök meg. Az ábrázolás helyes módjáról a Software Engineering with UML [5] könyv leírásai alapján inspirálódtam.

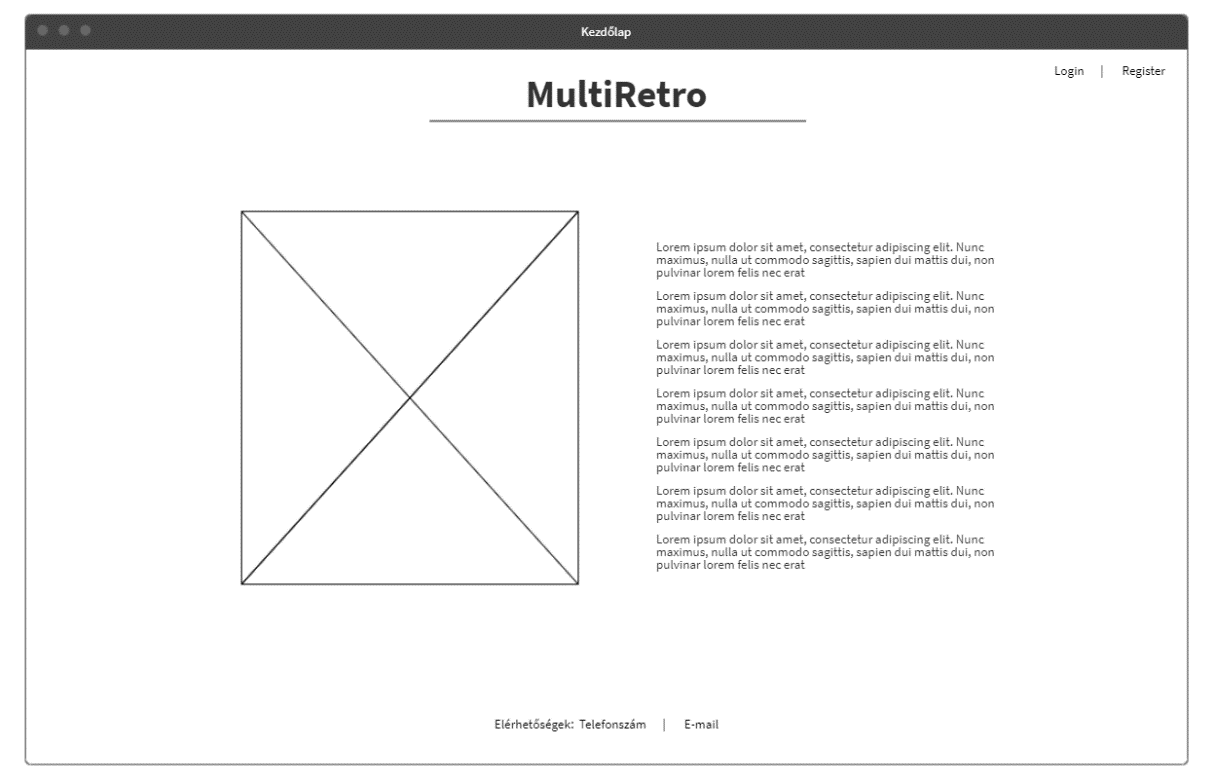
## 3.1 Fejlesztő környezet

A MultiRetro egy webes alkalmazás, melyet a reszponzív beállításoknak köszönhetően egyaránt kényelmes okostelefonon és számítógépen használni. Az implementáció Laravel PHP Framework [1] segítségével történt, Bootstrap [2] felel a megjelenésért, az adatbázishoz az SQLite [3] szolgáltatja az alapot.

A Laravel (8.x) telepítése Composer (2.0), Node.js (14.16), azon belül npm (6.14) segítségével történt, Visual Studio Code volt a fejlesztői környezet. A bejelentkezés/regisztráció/kilépés funkciók Laravel Fortify csomag segítségével lettek létrehozva.

A weblap Heroku [8] segítségével lett elérhetővé téve a személyes GitHub felhasználómon keresztül [9].

## 3.2 Drótváztervek

A MultiRetro webes alkalmazás hét darab fő nézetből áll: kezdőlap, profil, csapatok, megbeszélések, akciópontok, blogok, napló. A drótvázak elkészítése a MockFlow[[5]](#footnote-5) nevű ingyenes webes alkalmazással lett megvalósítva.

A *kezdőlap* felépítése a 23-as ábrán látható - főcím, egy kép és mellette a weblap rövid leírása. A felső jobb sarokban látható a bejelentkezés, regisztráció, a lap alján pedig az elérhetőségek.

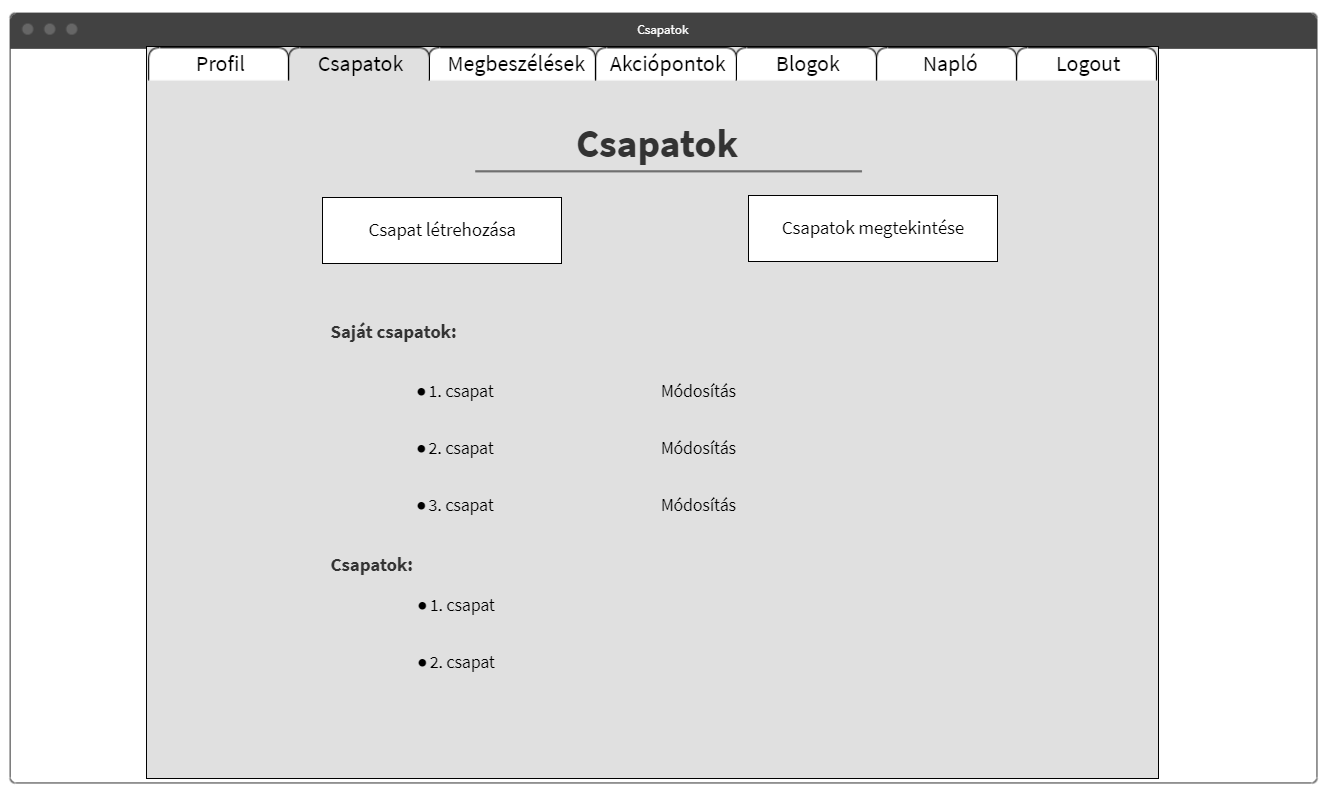
23. ábra: Drótvázterv - kezdőlap

A *profil* fülre rákattintva a felhasználó adatait láthatjuk, valamint egy akciógombot, aminek segítségével módosítani lehet ezeket az adatokat. (lásd: 24. ábra)



24. ábra: Drótvázterv - Profil

A 25-ös ábra jeleníti meg a *csapatok* fül tervét, ahol látható listázva, hogy milyen csapatokat hoztunk létre, valamint itt lehet módosítani is őket vagy újat létrehozni. A csapatok listázása gombra azért van szükség, hogy legyen lehetőség látni az összes csapatot, ne csak azokat, amiket mi hoztunk létre vagy tagjai vagyunk.



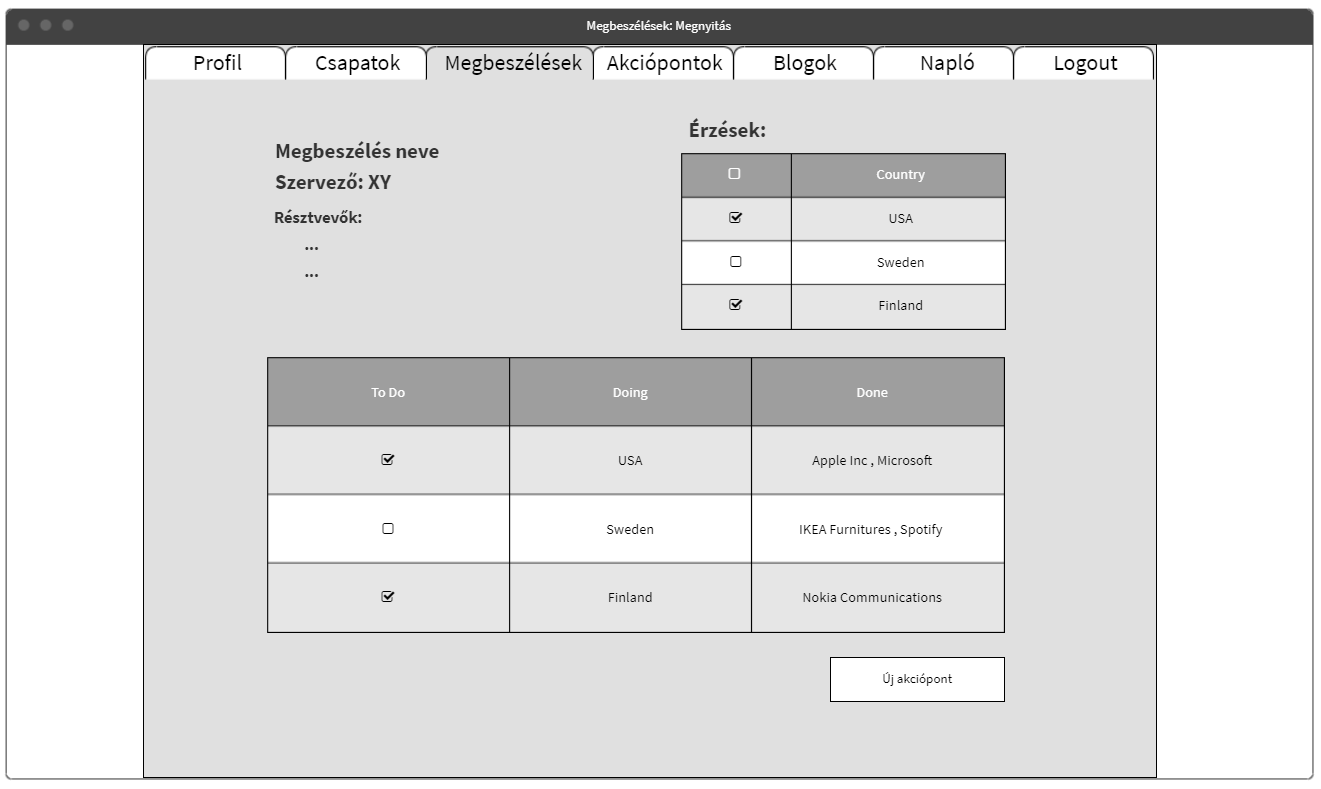
25. ábra: Drótvázterv - Csapatok

A *megbeszélés* részhez két terv is készült. Az első az alapnézet, ami a 26. ábrán látható, azokat a megbeszéléseket listázza ki, amikre meg vagyunk hívva. Ha mi hoztuk létre a megbeszélést, akkor itt tudjuk módosítani azt. Van lehetőség az adatbázisban szerepelő összes megbeszélés kilistázására is, mely segít akár az új megbeszélések létrehozásánál – időpont ütközés elkerülése végett.

A 27. ábrán látható nézet akkor jelenik meg, ha az alapnézeten rákattintunk a megnyitás gombra bármelyik megbeszélés esetén. Az elrendezés és a végleges funkciók még nem voltak meghatározva a drótváz készítésénél, viszont már körvonalazódott, hogy egy Kanban tábla, egy rövid űrlap és a megbeszélés adatainak leírása a részei lesznek.

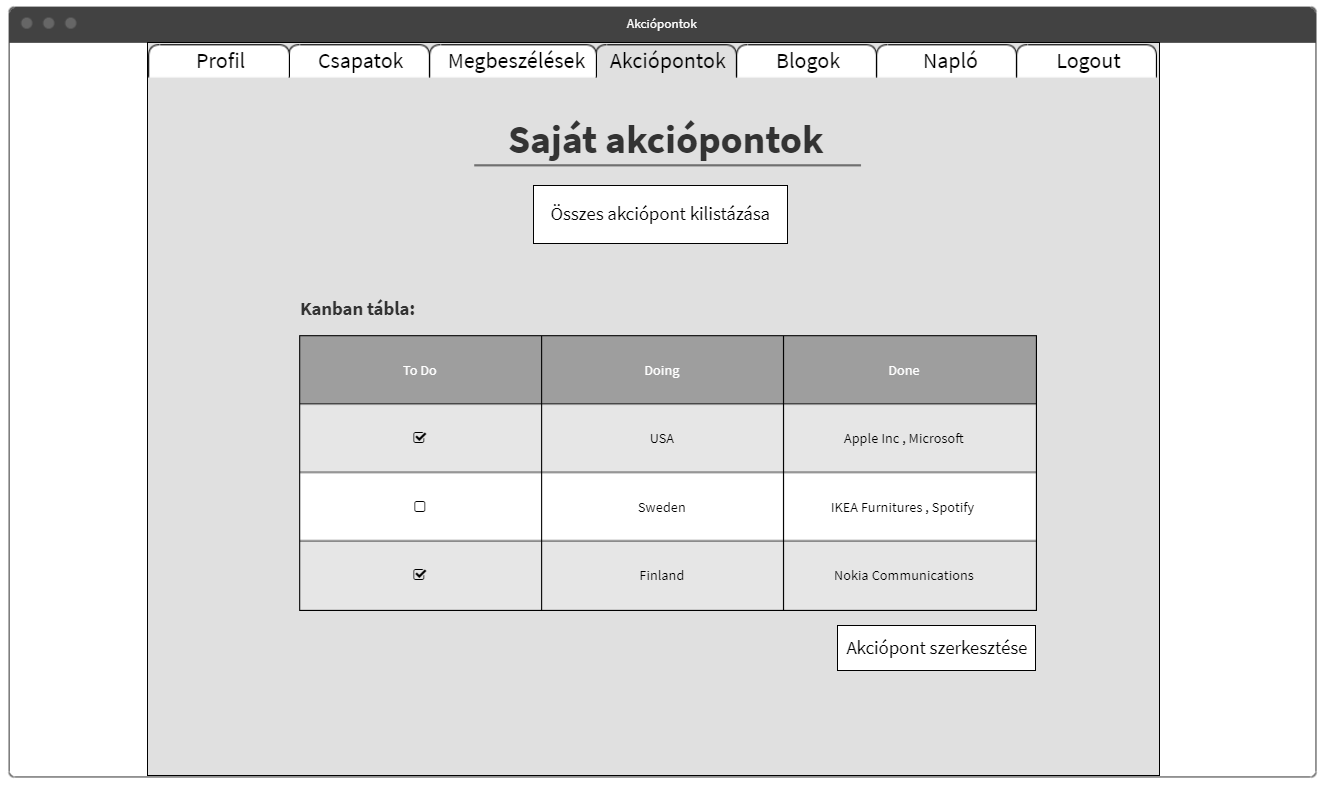


26. ábra: Drótvázterv – Megbeszélések alapnézete



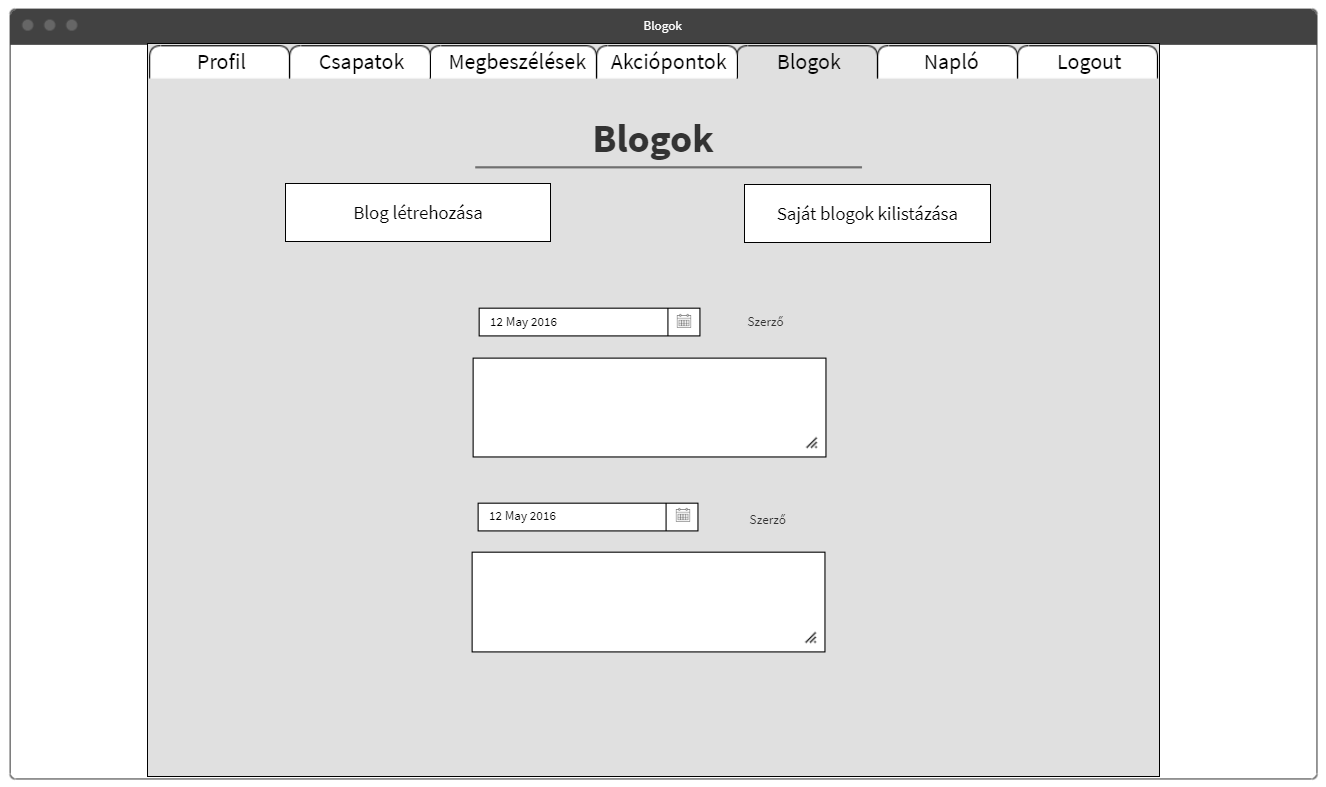
27. ábra: Drótvázterv – Megbeszélés részletei

Az *akciópontok* fül alatt a saját akciópontjaink jelennek meg Kanban táblába foglalva, ahogy az a 28-as ábrán látható. Az összes akciópont kilistázásánál megbeszélések szerint csoportosítva jelenik meg az összes akciópont ugyanúgy Kanban táblába rendezve.



28. ábra: Drótvázterv - Akciópontok

A *blogok* fül felel a blogbejegyzésekért, melyet a 29-es ábrán láthatunk. Itt, a többi menüponttal ellentétben nem a saját blogjainkat listázza ki alapból, hanem mindenkiét. A saját blogok a második gombra kattintva jelennek meg, itt tudjuk módosítani is őket. Természetesen lehetőség van új blogbejegyzés létrehozására, mely az első gombra kattintva válik elérhetővé.



29. ábra: Drótvázterv - Blogok

A *napló* drótvázterv csupán pár linket és adatokat tartalmaz, ahogy azt a 30-as ábrán láthatjuk. A megbeszélések neve a link, amivel a megbeszélés összesített átlag eredményei és a hozzá tartozó akciópontok és plusz-mínusz kártyák elmentett adatai jeleníthetők meg.



30. ábra: Drótvázterv - Napló

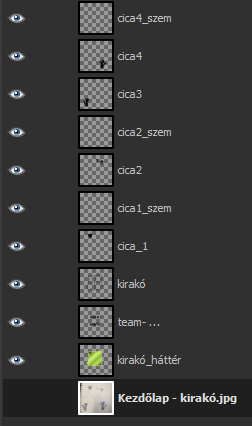
## 3.3 Grafika

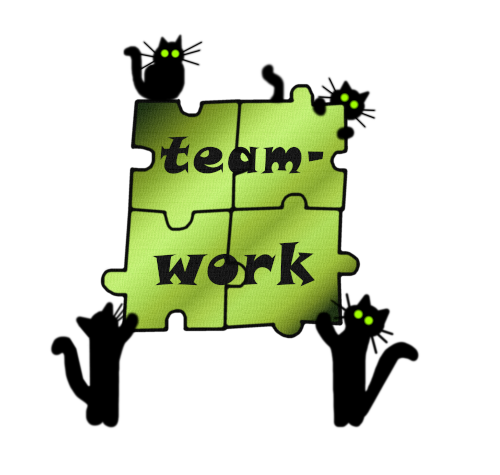
### 3.3.1 Kezdőlapon szereplő kép – cat\_puzzle.png

A kezdőlapon szereplő képet – 31. ábra - Gimp[[6]](#footnote-6) (2.10.12) grafikus program segítségével készítettem. Először papíron rajzoltam meg a vázlatot, majd ez alapján valósítottam meg számítógépen az eszközök használatával.

Az alap koncepció az volt, hogy a csapatmunkát jelenítsem meg egy egyszerű képen. Nem szerettem volna embereket ábrázolni, így döntöttem egyszerű fekete macskák mellett.

31. ábra: cat\_puzzle - vázlat





32. ábra: cat\_puzzle 33. ábra: cat\_puzzle - Gimp rétegek

Különböző réteget hoztam létre minden fontosabb elemnek a könnyebb szerkeszthetőség érdekében (33 .ábra). Nem használtam előre definiált formákat, mindent szabad kézzel rajzoltam az ecsetek stílusát és méretét változtatva. A teamwork felirat Snap ITC betűstílusú, a macskákon Gauss-elmosást, a kirakón vászon szűrőt alkalmaztam. A végeredmény a 32-es ábrán látható.

### 3.3.2 Időjárás-jelentés ikonok

A megbeszélés lap egy hangulatjelentés retrospektív technika segédeszközeit is biztosítja. A különböző időjárást jelképező ikonokat látva a felhasználó kiválaszthatja azt, amely az ő hangulatát tükrözi a csapatmunka adott szempontjait illetően. Ezt a négy darab képet mutatja be a 34-es ábra:

34. ábra: Időjárás jelentés ikonok

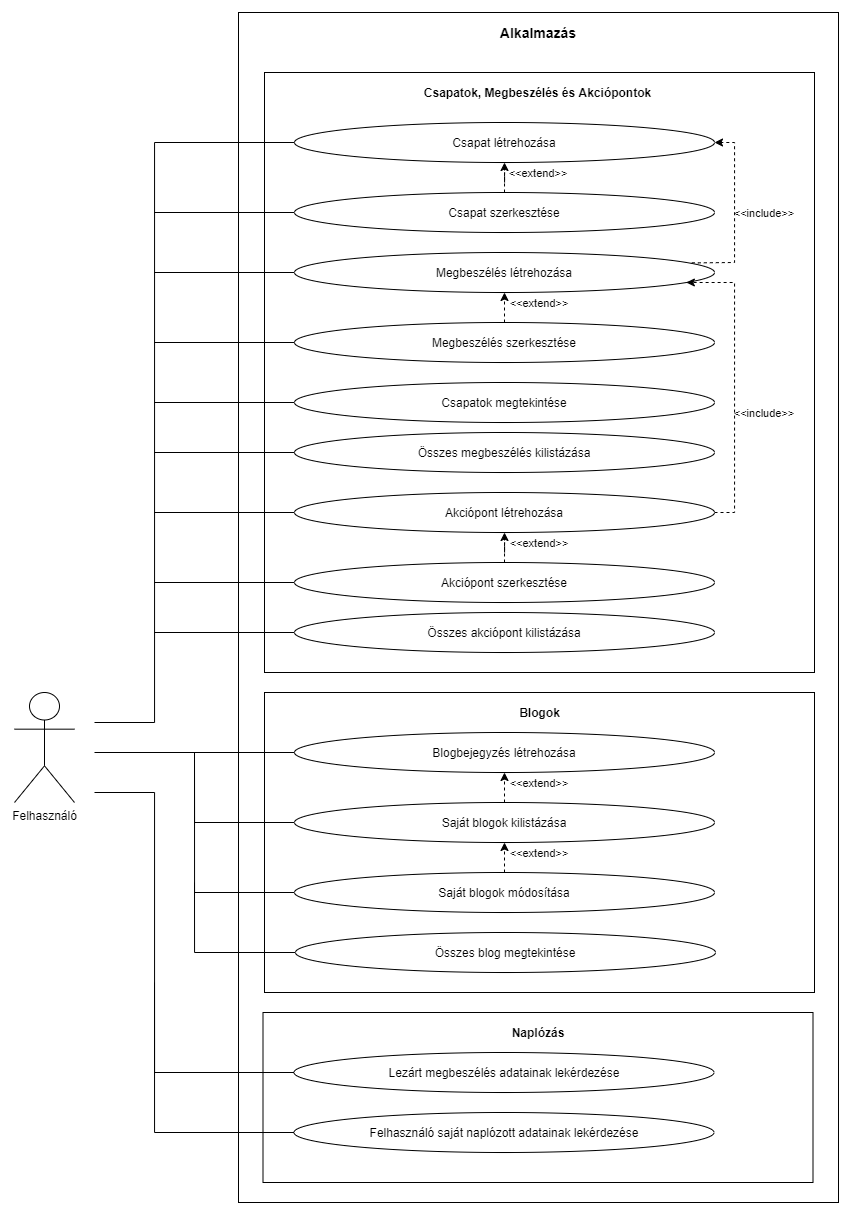
* sunny.png
* cloudy.png
* rainy.png
* stormy.png

Ezeket szintén a Gimp [12] nevű rajzolóprogrammal készítettem. Az alakzatokat kijelölések egymásba ágyazásával és háttérszín kitöltéssel vagy ecsettel készítettem. A weblapba a lekicsinyített megfelelőjüket illesztettem be.

## 3.4 Használati esetek

A használati esetek három fő csoportba sorolhatók, ahogy az a 35-ös ábrán látható:

* *Csapatok, megbeszélés és akciópontok:* bemutatja a három kategória kapcsolódását, valamint azt is, hogyan lehet külön-külön kezelni őket.
* *Blogok:* mivel ez egy külön funkció, így csak a blogok kezelését tartalmazza.
* *Naplózás:* ebbe a csoportba tartozik a lezárt megbeszélések lementett adatainak megtekintése, illetve a felhasználó saját naplózott adatainak megtekintése.



35. ábra: Használati eset diagram

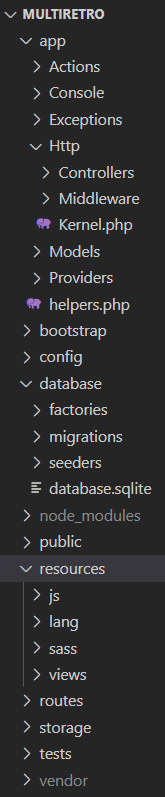
## C:\Users\Gábor\AppData\Local\Microsoft\Windows\INetCache\Content.Word\MVC diagram.png3.5 Architektúra

Az alkalmazás meghatározó architektúrája a 36-os ábrán látható **M**odel-**V**iew-**C**ontroller architektúra.

A *Model* egy valós példány vagy objektum reprezentációja a kódban. A *View* azt a felületet jelenti, amelyen keresztül a felhasználó interakcióba lép az alkalmazással. Amikor egy felhasználó műveletet hajt végre, a *Controller* kezeli a műveletet, és szükség esetén frissíti a *Model*-t.

36. ábra: MVC diagram

**Laravel request működése**

Ha új Laravel projektet hoztunk létre, és a szükséges bővítményeket is telepítettük, akkor a 37-es ábrán látható módon épül fel a mappaszerkezetünk.

A *routes* mappában található a web.php nevű fájl, itt lehet definiálni, hogy az URL útvonalak melyik controllerekhez kapcsolódjanak, és azon belül melyik függvényt hívják meg.

Az *app/Http/Controllers* mappában találhatók meg az előbb említett controllerek. Ezeken belül találhatók rendszerezve a modelekhez tartozó függvények.

A *database/migrations* mappában vannak a táblák meghatározásai, a *database/seeders* mappában az automatikusan feltölthető tesztadatok, és az *app/Models*-en belül vannak a migration-ökhöz tartozó modellek definíciói. Az SQLite adatbázis fájl a *database* mappán belül található database.sqlite néven.

A *resources/views* mappán belül találhatók a bladek, amik a megjelenítésért felelnek.

A *tests* mappában találhatók az automatikus tesztek.

A Laravel architektúra további felépítéséről részletesen szó esik a dokumentációban [1] az „Architecture Concepts” menüpont alatt.

37. ábra: Mappaszerkezet

## 3.6 Adatbázis

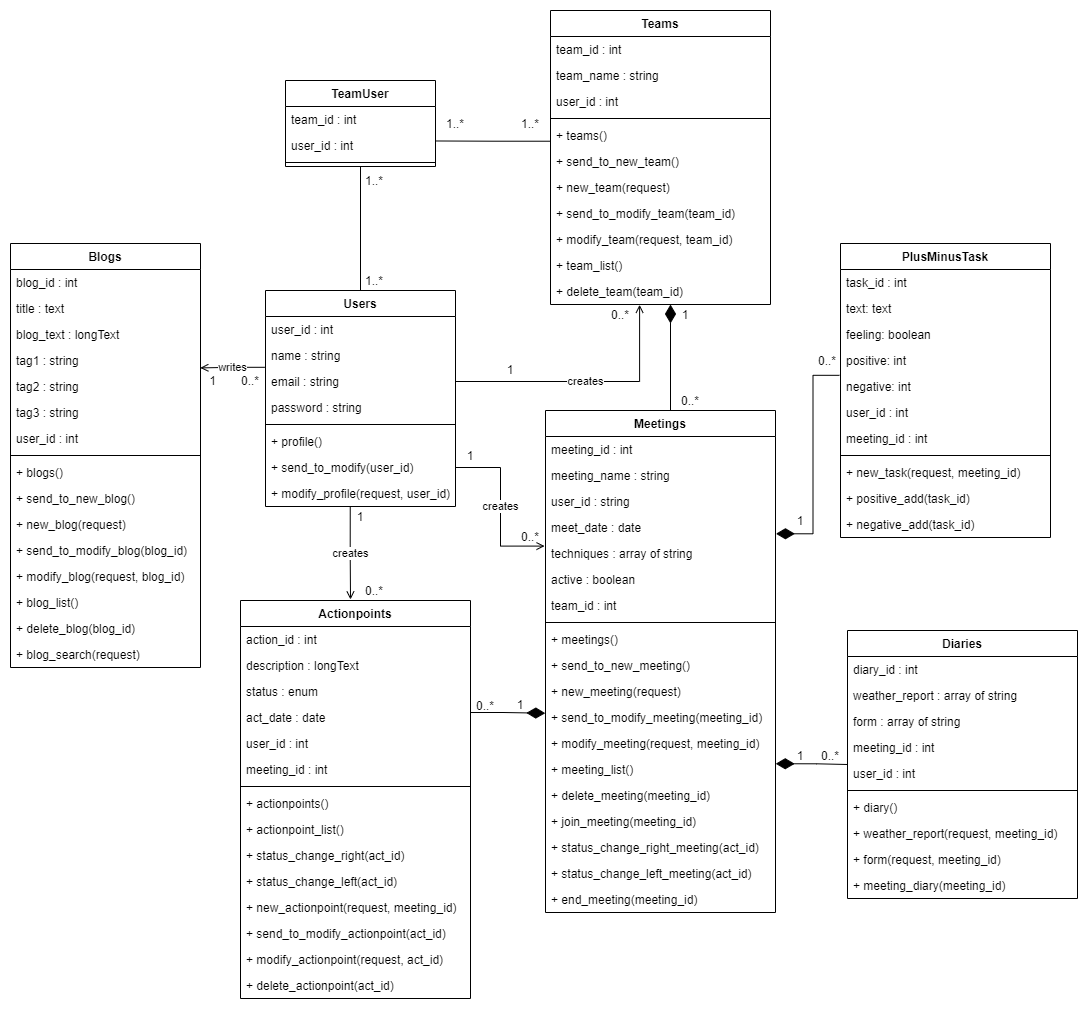
### 3.6.1 Az adatbázis felépítése

Az adatbázis a 38. ábrán látható módon épül fel, központjában a felhasználó (users) tábla áll, amihez az összes többi csatlakozik. Továbbá az ábra szemlélteti a különböző táblákhoz tartozó adattagokat és a hozzájuk tartozó metódusokat. A fő táblák mellett található egy összekötő tábla, ami az n → n kapcsolat miatt szükséges, hiszen segítségével sokkal hatékonyabb a kommunikáció a felhasználó és csapat táblák között.

A fő táblák:

* Users: felhasználók adatait tárolja
* Teams: csapatok adatait tárolja
* Meetings: megbeszélés adatait tárolja
* Actionpoints: a megbeszélés közben létrehozott akciópontokat tárolja
* PlusMinusTasks: a megbeszélés közben létrehozott plusz-mínusz kártyákat tárolja
* Blogs: blogok adatait tárolja
* Diaries: megbeszélések eredményeit tárolja

Összekötő tábla: TeamUser.

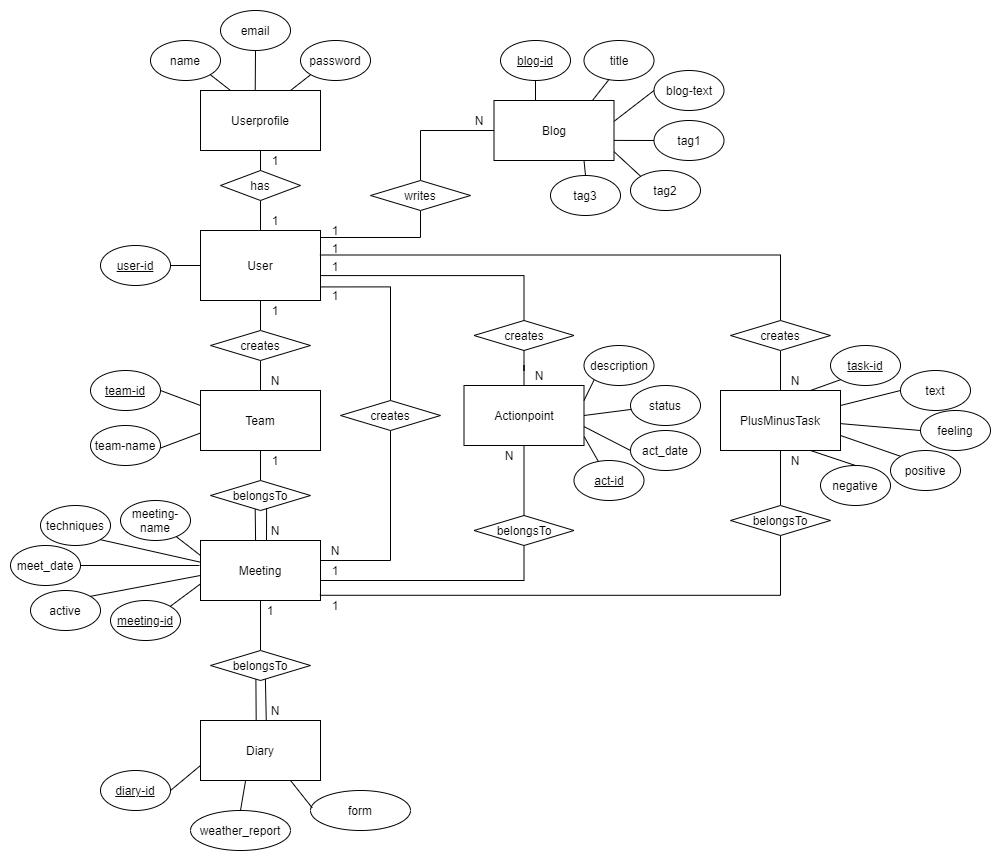


38. ábra: Adatbázis osztálydiagram

Az ábrán látható osztálydiagram értelmezése egyedi, ugyanis az adattagok a migration-ből, a függvények pedig a hozzájuk tartozó modelek-ből lettek kiemelve.

### 3.6.2 Egyed-kapcsolat diagram

A különböző objektumok egymással való kapcsolatát a 39-es ábrán látható egyed-kapcsolat diagram ábrázolja.



39. ábra: Egyed-kapcsolat diagram

### 3.6.3 Migration és Model

A Laravel által támogatott adatbázis rendszer része az ún. migration, amely sémákkal egyszerűbbé teszi a táblák létrehozását és módosítását.

A MultiRetro webes alkalmazáshoz az alábbi migration-öket hoztam létre: users, teams, meetings, team\_user, actionpoints, plus\_minus\_tasks, blogs, diaries.

A meetings migration törzse az alábbi módon néz ki:

**Schema**::**create**('meetings', function (**Blueprint** $table) {

            $table->**id**();

            $table->**string**('name');

            $table->**unsignedBigInteger**('owner');

            $table->**date**('meet\_date');

            $table->**json**('techniques')->**nullable**();

            $table->**boolean**('active')->**default**('true');

            $table->**unsignedBigInteger**('team\_id');

            $table->**timestamps**();

            $table->**foreign**('owner')->**references**('id')->**on**('users');

            $table->**foreign**('team\_id')->**references**('id')->**on**('teams');

        });

Első lépésként a meetings tábla oszlopait típussal együtt definiáljuk, majd beállítjuk a referenciákat. Itt például az owner adattagról elmondjuk, hogy a users táblában az id-nak felel meg. Továbbá lehetőség van különböző bővítésekre, megadni a default értékeket vagy beállítani, hogy egy oszlop nullable legyen, azaz nem muszáj adatot tartalmaznia.

A táblák összekapcsolását az App\Models mappán belül valósítottam meg. Minden fő tábla mellé létrehoztam egy modellt: User, Team, Meeting, Actionpoint, Blog, Diary és PlusMinusTask.

Az alábbi kapcsolatok valósultak meg:

* User.php:
  + teams(): belongsToMany kapcsolat a Team osztállyal → egy felhasználó több csapathoz is tartozhat [n → n]
  + owner\_teams(): hasMany kapcsolat a Team osztállyal → egy felhasználónak több csapata is lehet [1 → n]
  + owner\_meetings(): hasMany kapcsolat a Meeting osztállyal → egy felhasználónak több megbeszélése is lehet [1 → n]
  + owner\_actionpoints(): hasMany kapcsolat az Actionpoint osztállyal → egy felhasználónak több akciópontja is lehet [1 → n]
  + owner\_blogs(): hasMany kapcsolat a Blog osztállyal → egy felhasználónak több blogbejegyzése is lehet [1 → n]
  + owner\_tasks(): hasMany kapcsolat a PlusMinusTask osztállyal → egy felhasználónak több plusz-mínusz kártyája is lehet [1 → n]
  + owner\_diaries(): hasMany kapcsolat a Diary osztállyal → egy felhasználónak több naplózott sora is lehet [1 → n]
* Team.php:
  + users(): belongsToMany kapcsolat a User osztállyal → egy csapat több felhasználóhoz tartozhat [n → n]
  + team\_owner(): belongsTo kapcsolat a User osztállyal → egy csapatnak csak egy tulajdonosa lehet [1 → 1]
  + meetings(): hasMany kapcsolat a Meeting osztállyal → egy csapatnak több megbeszélése is lehet [1 → n]

Az alábbi kódrészlet mutatja be, hogyan néz ki a kapcsolatok kialakítása a Team modelben:

*//egy csapat több felhasználóhoz tartozhat: n->n*

    public function **users**() {

        return $this->**belongsToMany**(**User**::class)->**withTimestamps**();

    }

*//egy csapatnak csak egy tulajdonosa lehet: 1->1*

    public function **team\_owner**() {

        return $this->**belongsTo**(**User**::class, 'user\_id', 'id');

    }

*//egy csapatnak több meetingje is lehet: 1->n*

    public function **meetings**() {

        return $this->**hasMany**(**Meeting**::class);

    }

* Meeting.php:
  + team(): belongsTo kapcsolat a Team osztállyal → egy megbeszélés csak egy csoporthoz tartozhat [1 → 1]
  + meet\_owner(): belongsTo kapcsolat a User osztállyal → egy megbeszélés csak egy felhasználóhoz tartozhat [1 → 1]
  + actionpoints(): hasMany kapcsolat az Actionpoint táblával → egy megbeszéléshez több akciópont tartozhat [1 → n]
  + plus\_minus\_tasks(): hasMany kapcsolat a PlusMinusTask osztállyal → egy megbeszéléshez több plusz-mínusz feladat tartozhat [1 → n]
  + diaries(): hasMany kapcsolat a Diary osztállyal → egy megbeszéléshez több naplózott sor is tartozhat [1 → n]
* Actionpoint.php:
  + action\_owner(): belongsTo kapcsolat a User osztállyal → egy akciópont xsak egy megbeszéléshez tartozhat [1 → 1]
  + meeting\_act(): belongsTo kapcsolat a Meeting osztállyal → egy akciópont csak egy megbeszéléshez tartozhat [1 → 1]
* PlusMinusTask.php:
  + task\_owner(): belongsTo kapcsolat a User osztállyal → egy feladatnak csak egy tulajdonosa lehet [1 → 1]
  + meeting\_task(): belongsTo kapcsolat a Meeting osztállyal → egy feladat csak egy megbeszéléshez tartozhat [1 → 1]
* Blog.php:
  + blog\_owner(): belongsTo kapcsolat a User osztállyal → egy blognak csak egy tulajdonosa lehet [1 → 1]
* Diary.php:
  + diary\_owner(): belongsTo kapcsolat a User osztállyal → egy naplózott sornak csak egy tulajdonosa lehet [1 → 1]
  + meeting\_diary(): belongsTo kapcsolat a Meeting osztállyal → egy naplózott sor csak egy megbeszéléshez tartozhat [1 → 1]

### 3.6.4 Seeding

Az ún. seeder osztályokkal lehetőség van arra, hogy az adatbázist teszt adatokkal töltsük fel, így elősegítve a funkciók kipróbálását. Van egy automatikusan implementált osztály, a DatabaseSeeder, itt kell meghívni azokat a seedereket, amiket szeretnénk lefuttatni a migration frissítésével.

A MultiRetro az alábbi seeder-eket használja:

* UserSeeder:

**DB**::**table**('users')->**insert**([

            'name' => 'Admin Admin',

            'email' => 'admin@admin.hu',

            'password' => **Hash**::**make**('adminadmin'),

            'created\_at' => **now**(),

            'updated\_at' => **now**(),

        ]);

* TeamSeeder
* UserTeamSeeder

**DB**::**table**('team\_user')->**insert**([

            'user\_id' => '1',

            'team\_id' => '4',

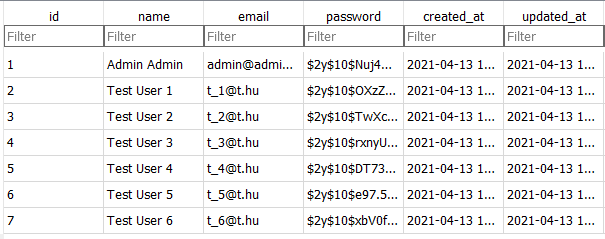
            'created\_at' => **now**(),

            'updated\_at' => **now**(),

        ]);

* MeetingSeeder
* ActionpointSeeder
* BlogSeeder
* DiarySeeder
* PlusMinusTaskSeeder

A seeder ellenőrzésére egy egyszerű adatbázis-kezelő program volt használva: DB Browser for SQLite.[[7]](#footnote-7) A felület a 40-es ábrán látható.



40. ábra: DB Browser - users table

## 3.7 Controllers and routes – kontrollerek és útvonalak

A Laravel a controllerek segítségével biztosít lehetőséget az elérési útvonalak rendszerezésére. Ez rendkívül hasznos, hiszen sokkal átláthatóbb a kód, és könnyebb szerkeszteni is.

A MultiRetro alkalmazásban a következő controllerek találhatók:

* UserProfileController
* TeamController
* MeetingController
* ActionpointController
* DiaryController
* PlusMinusTaskController
* BlogController

### 3.7.1 UserProfileController

A felhasználói profil rendszerezésére szolgál ez a controller, az alábbi metódusokat hoztam létre:

* profile(): a userprofile nézetre dob
* send\_to\_modify(user\_id): a felhasználó azonosítójával együtt elküld a modify\_userprofile nézetre
* modify\_profile(request, user\_id): ellenőrzés után lementi a módosításokat és visszaküld a userprofile nézetre

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//PROFIL*

**Route**::**get**('/userprofile', [**UserProfileController**::class,'profile'])->**name**('userProfile');

*//profil módosítása*

**Route**::**get**('/userprofile/{user\_id}', [**UserProfileController**::class, 'send\_to\_modify'])->**name**('sendToModifyUser');

**Route**::**post**('/userprofile/{user\_id}', [**UserProfileController**::class, 'modify\_profile'])->**name**('modifyUserProfile');

### 3.7.2 TeamController

A csapatok rendszerezésére szolgál ez a controller, az alábbi metódusokat biztosítja:

* teams(): a teams nézetre dob, itt definiálom a teams és tagok tömböket, melyekre a különböző listázásoknál van szükség
* send\_to\_new\_team(): a modify\_team nézetre visz az users változóval együtt
* new\_team(request): az adatok ellenőrzése után lementi az új csapatot az adatbázisba, majd a teams nézetre dob vissza
* send\_to\_modify\_team(team\_id): a szerkeszteni kívánt csapat azonosítójával és a users változóval együtt elküld a modify\_team nézetre
* modify\_team(request, team\_id): az adatok validálása után lementi a változtatásokat az adatbázisba, majd visszaküld a teams nézetre
* team\_list(): a team\_list nézetre küld a teams változóval együtt
* delete\_team(team\_id): törli a kiválasztott csapatot, majd visszadob a teams nézetre

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//CSAPATOK*

**Route**::**get**('/teams', [**TeamController**::class,'teams'])->**name**('teams');

*//új csapat*

**Route**::**get**('/new\_team', [**TeamController**::class,'send\_to\_new\_team'])->**name**('sendToNewTeam');

**Route**::**post**('/new\_team', [**TeamController**::class,'new\_team'])->**name**('newTeam');

*//csapat módosítása*

**Route**::**get**('/teams/{team\_id}', [**TeamController**::class,'send\_to\_modify\_team'])->**name**('sendToModifyTeam');

**Route**::**post**('/teams/{team\_id}', [**TeamController**::class,'modify\_team'])->**name**('modifyTeam');

*//csapat listázása*

**Route**::**get**('/team\_list', [**TeamController**::class,'team\_list'])->**name**('teamList');

*//csapat törlése*

**Route**::**get**('/team\_list/{team\_id}', [**TeamController**::class,'delete\_team'])->**name**('deleteTeam');

### 3.7.3 MeetingController

A megbeszélések függvényei a MeetingController-ben találhatóak meg. Ezek a következők:

* meetings(): a meetings nézetre dob, itt van definiálva a meetings, teams, all\_meet változókat, amikre a megbeszélések listázásánál van szükség
* send\_to\_new\_meeting(): a modify\_meeting nézetre visz a teams változóval együtt
* new\_meeting(request): az adatok ellenőrzése után lementi az új megbeszélést az adatbázisba és visszadob a meetings nézetre
* send\_to\_modify\_meeting(meeting\_id): a modify\_meeting nézetre dob a változtatni kívánt megbeszélés azonosítóval együtt
* modify\_meeting(request, meeting\_id): a változtatások ellenőrzése után frissíti az adatbázist és visszadob a meetings nézetre, ahogy az az alábbi kódrészleten látható

    public function **modify\_meeting**(**Request** $request, $meeting\_id) {

        $meeting = **Meeting**::**where**('id', $meeting\_id)->**firstOrFail**();

        $validated = $request->**validate**([

            'name' => 'required|max:255',

            'meet\_date' => 'required|date|after:yesterday',

            'team' => 'required',

        ]);

        $meeting->**update**($validated);

        $meeting->team\_id = $request->**input**('team');

        $techniques = $meeting->techniques;

        $techniques['0'] = $request->**input**('report');

        $techniques['1'] = $request->**input**('plus\_minus');

        $techniques['2'] = $request->**input**('form');

        $meeting->techniques = $techniques;

        $rec = $request->**user**()->**owner\_meetings**()->**save**($meeting);

        return **redirect**()->**route**('meetings', ['meeting\_id' => $meeting->$meeting\_id]);

    }

* meeting\_list(): a meeting\_list nézetre visz a meetings változóval együtt
* delete\_meeting(meeting\_id): törli a kiválasztott megbeszélést a hozzá tartozó adatokkal együtt, majd visszadob a meetings nézetre
* join\_meeting(meeting\_id): a paraméterül megadott meeting\_id-hoz tartozó megbeszélést tölti be a join\_meeting nézeten
* status\_change\_right\_meeting(act\_id): a megbeszéléshez tartozó Kanban-táblában az akciókártyák tetején lévő jobb nyilakra kattintva hajtódik végre, egy oszloppal jobbra küldi a kártyát
* status\_change\_left\_meeting(act\_id): a megbeszéléshez tartozó Kanban-táblában az akciókártyák tetején lévő bal nyilakra kattintva hajtódik végre, egy oszloppal balra küldi a kártyát
* end\_meeting(meeting\_id): az adott megbeszélés lezárása, a meetings nézetre dob vissza

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//MEGBESZÉLÉS*

**Route**::**get**('/meetings', [**MeetingController**::class,'meetings'])->**name**('meetings');

*//új megbeszélés*

**Route**::**get**('/new\_meeting', [**MeetingController**::class,'send\_to\_new\_meeting'])->**name**('sendToNewMeeting');

**Route**::**post**('/new\_meeting', [**MeetingController**::class,'new\_meeting'])->**name**('newMeeting');

*//megbeszélés módosítása*

**Route**::**get**('/meetings/{meeting\_id}', [**MeetingController**::class,'send\_to\_modify\_meeting'])->**name**('sendToModifyMeeting');

**Route**::**post**('/meetings/{meeting\_id}', [**MeetingController**::class,'modify\_meeting'])->**name**('modifyMeeting');

*//megbeszélés listázása*

**Route**::**get**('/meeting\_list', [**MeetingController**::class,'meeting\_list'])->**name**('meetingList');

*//megbeszélés törlése*

**Route**::**get**('/meeting\_list/{meeting\_id}', [**MeetingController**::class,'delete\_meeting'])->**name**('deleteMeeting');

*//csatlakozás*

**Route**::**get**('/join\_meeting/{meeting\_id}', [**MeetingController**::class,'join\_meeting'])->**name**('joinMeeting');

*//megbeszélésen belül akciópont státusz módosítás*

**Route**::**get**('/right{act\_id}', [**MeetingController**::class,'status\_change\_right\_meeting'])->**name**('statusChangeRightMeeting');

**Route**::**get**('/left{act\_id}', [**MeetingController**::class,'status\_change\_left\_meeting'])->**name**('statusChangeLeftMeeting');

*//megbeszélés lezárása*

**Route**::**get**('/meeting\_end{meeting\_id}', [**MeetingController**::class,'end\_meeting'])->**name**('endMeeting');

### 3.7.4 ActionpointController

Az akciópontok műveleteiért felelős ez a controller. Ezek a következők:

* actionpoints(): az actionpoints nézetre visz az actionpoints változóval együtt, amire az akciópontok Kanban-táblába rendezésénél van szükség
* actionpoint\_list(): az actionpoint\_list nézetre visz az actionpoints és meetings változókkal együtt
* status\_change\_right(act\_id): az akciópontok státuszának módosításáért felel, egy oszloppal jobbra helyezi át a kártyát
* status\_change\_left(act\_id): az akciópontok státuszának módosításáért felel, egy oszloppal balra helyezi át a kártyát
* new\_actionpoint(request, meeting\_id): az adatok validálása után lementi az új akciópontot az adatbázisba
* send\_to\_modify\_actionpoint(act\_id): az akciópont azonosítójával együtt elküld a modify\_actionpoint nézetre
* modify\_actionpoint(request, act\_id): az adatok ellenőrzése után lementi a módosításokat az adatbázisba, majd visszadob az actionpoints nézetre
* delete\_actionpoint(act\_id): törli az adott akciópontot, majd visszadob az actionpoint nézetre

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//AKCIÓPONT*

**Route**::**get**('/actionpoints', [**ActionpointController**::class,'actionpoints'])->**name**('actionpoints');

*//csapat listázása*

**Route**::**get**('/actionpoint\_list', [**ActionpointController**::class,'actionpoint\_list'])->**name**('actionpointList');

*//státusz módosítás*

**Route**::**get**('/actionpoints/right{act\_id}', [**ActionpointController**::class,'status\_change\_right'])->**name**('statusChangeRight');

**Route**::**get**('/actionpoints/left{act\_id}', [**ActionpointController**::class,'status\_change\_left'])->**name**('statusChangeLeft');

*//új akciópont*

**Route**::**get**('/new\_actionpoint{meeting\_id}', [**ActionpointController**::class,'new\_actionpoint'])->**name**('newActionpoint');

*//akciópont módosítása*

**Route**::**get**('/actionpoints/{act\_id}', [**ActionpointController**::class,'send\_to\_modify\_actionpoint'])->**name**('sendToModifyActionpoint');

**Route**::**post**('/actionpoints/{act\_id}', [**ActionpointController**::class,'modify\_actionpoint'])->**name**('modifyActionpoint');

*//akciópont törlése*

**Route**::**get**('/actionpoint\_delete\_{act\_id}', [**ActionpointController**::class,'delete\_actionpoint'])->**name**('deleteActionpoint');

### 3.7.5 DiaryController

A napló controlleréhez tartozó függvények a következők:

* diary(): a diary nézetre visz a diaries változóval együtt
* weather\_report(request, meeting\_id): ellenőrzés után lementi az adatbázisba a beküldött időjárás jelentést
* form(request, meeting\_id): ellenőrzés után lementi az adatbázisba a beküldött űrlapot
* meeting\_diary(meeting\_id): a paraméterül megadott megbeszélés azonosítóhoz tartozó naplózott adatokat jeleníti meg a meeting\_diary nézeten

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//NAPLÓ*

**Route**::**get**('/diary', [**DiaryController**::class,'diary'])->**name**('diary');

*//időjárás jelentés*

**Route**::**get**('/diary/weather\_report{meeting\_id}', [**DiaryController**::class,'weather\_report'])->**name**('weatherReport');

*//űrlap*

**Route**::**get**('/diary/form{meeting\_id}', [**DiaryController**::class,'form'])->**name**('form');

*//megbeszélés naplózása*

**Route**::**get**('/meeting\_diary{meeting\_id}', [**DiaryController**::class,'meeting\_diary'])->**name**('meetingDiary');

### 3.7.6 PlusMinusTaskController

A megbeszélés egyik választható technikája a plusz-mínusz táblázat. Az ehhez kapcsoló kártyák utasításai a következők:

* new\_task(request, meeting\_id): az adatok validálása után lementi az adatbázisba a plusz-mínusz kártyát
* positive\_add(task\_id): az adott kártyához tartozó positive adattaghoz hozzáad egyet
* negative\_add(task\_id): az adott kártyához tartozó negative adattaghoz hozzáad egyet

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//PLUSZ-MÍNUSZ FELADAT*

*//új plus-mínusz feladat*

**Route**::**get**('/new\_plus\_minus{meeting\_id}', [**PlusMinusTaskController**::class,'new\_task'])->**name**('newPlusMinusTask');

*//pozitív hozzáadás*

**Route**::**get**('/add\_positive{task\_id}', [**PlusMinusTaskController**::class,'positive\_add'])->**name**('positiveAdd');

*//negatív hozzáadás*

**Route**::**get**('/add\_negative{task\_id}', [**PlusMinusTaskController**::class,'negative\_add'])->**name**('negativeAdd');

### 3.7.7 BlogController

A blogokért felelős controllerhez a következő függvények tartoznak:

* blogs(): a blogs nézetre visz a blogs változóval együtt
* send\_to\_new\_blog(): a modify\_blog nézetre visz
* new\_blog(request): az adatok ellenőrzése után lementi az adatbázisba a blogot, majd a blog\_list nézetre visz
* send\_to\_modify\_blog(blog\_id): a paraméterül megadott blog azonosítóval együtt a modify\_blog nézetre visz
* modify\_blog(request, blog\_id): az adatok ellenőrzése után módosítja a blogot az adatbázisban, majd a blog\_list nézetre visz
* blog\_list(): a blog\_list nézetre visz a blogs változóval együtt
* delete\_blog(blog\_id): törli a megadott blogot, majd a blog\_list nézetre visz
* blog\_search(request): a megadott címke alapján szűri a blogbejegyzéseket a blogs változó segítségével

A routes/web.php-ban a következő útvonalak azonosak a fentiekkel:

*//BLOG*

**Route**::**get**('/blogs', [**BlogController**::class,'blogs'])->**name**('blogs');

*//új blog*

**Route**::**get**('/new\_blog', [**BlogController**::class,'send\_to\_new\_blog'])->**name**('sendToNewBlog');

**Route**::**post**('/new\_blog', [**BlogController**::class,'new\_blog'])->**name**('newBlog');

*//blog módosítása*

**Route**::**get**('/blogs/{blog\_id}', [**BlogController**::class,'send\_to\_modify\_blog'])->**name**('sendToModifyBlog');

**Route**::**post**('/blogs/{blog\_id}', [**BlogController**::class,'modify\_blog'])->**name**('modifyBlog');

*//blog listázása*

**Route**::**get**('/blog\_list', [**BlogController**::class,'blog\_list'])->**name**('blogList');

*//blog törlése*

**Route**::**get**('/delete\_blog{blog\_id}', [**BlogController**::class,'delete\_blog'])->**name**('deleteBlog');

*//blog keresés*

**Route**::**get**('/blog\_search', [**BlogController**::class,'blog\_search'])->**name**('blogSearch');

## 3.8 Views - nézetek

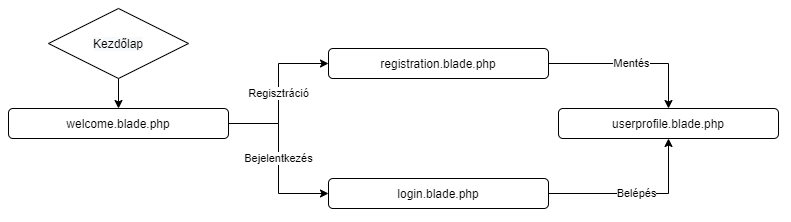
A webes alkalmazás képernyőn való megjelenéséért a nézetek felelnek. Ez Laravelben blade-ek használatával valósul meg. Az átláthatóság érdekében mappába rendeztem őket, igyekeztem a controllerek csoportosítását alapul venni. A resources/views mappán belül a következő mappák szerepelnek: auth, layouts, profile, team, meeting, actionpoint, diary, blog.

**Views**

A gyökérmappában csupán a welcome blade szerepel, ez jelenik meg elsőként a weblap megnyitásánál. Regisztráció vagy bejelentkezés után a kezdőlap szerepet kapja meg.

**Auth**

Ebben a mappában találhatóak a hitelesítésért felelős blade-ek, melyeket a 41-es folyamatábrán ábrázoltam. A bejelentkezésre kattintva jelenik meg a login blade, regisztrációra kattintva a register blade. Előbbinél a belépés gombra kattintva, utóbbinál a mentés gombra kattintva kerülünk át a userprofile blade-re, másnéven a profil nézetre.



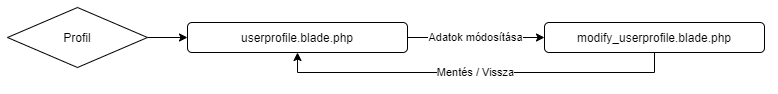
41. ábra: A kezdőlapért és hitelesítésért felelős nézetek

**Layouts**

Itt található az app.blade.php, ami lényegében meg van hívva minden egyes blade-ben, ugyanis ez felel a fejlécért.

**Profile**

A profil mappa tartalmazza a userprofile és modify\_userprofile blade-eket, melyek a 42-es ábrán láthatók. Előbbi felel a profil oldalért, míg utóbbi az adatok módosítása gombra kattintva jelenik meg.

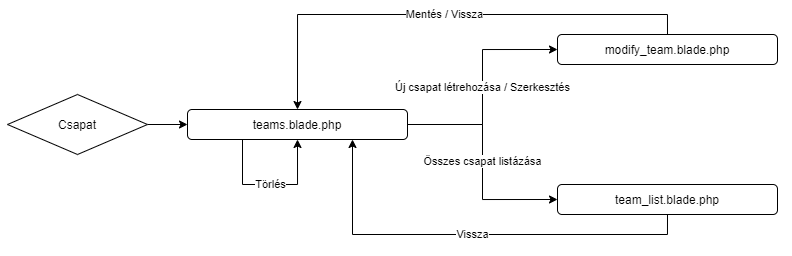


42. ábra: Profilhoz tartozó nézetek

**Team**

Három darab blade található a csapaton belül (lásd: 43. ábra):

* teams.blade.php: ez a nézet felel a csapat oldalért, minden művelet után ide kerülünk vissza
* modify\_team.blade.php: az ide mutató útvonaltól függ, hogy új csapat létrehozásának vagy csapat módosításának a nézete jelenik meg
* team\_list.blade.php: megjeleníti az összes csapatot listába rendezve

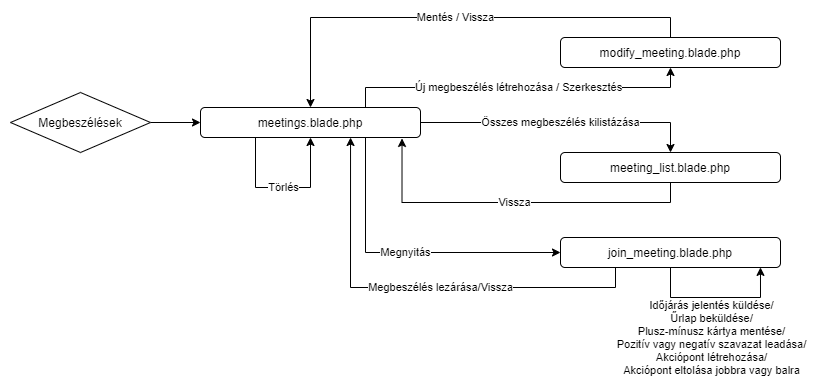


43. ábra: Csapathoz tartozó nézetek

**Meeting**

Az alábbi nézetek találhatók itt (lásd: 44. ábra):

* meetings.blade.php: ez a nézet felel a megbeszélés oldalért; megbeszélés létrehozása, módosítása, törlése után ide kerülünk vissza
* modify\_meeting.blade.php: az ide mutató útvonaltól függ, hogy új megbeszélés létrehozásának vagy megbeszélés módosításának a nézete jelenik meg
* meeting\_list.blade.php: megjeleníti az összes megbeszélést listába rendezve
* join\_meeting.blade.php: a megnyitás gombra kattintva kerülünk ide, itt zajlik a különböző retrospektívek használata: időjárás jelentés, kérdőív, plusz-mínusz feladat. Az akciópontok létrehozásának helye és a Kanban tábla is itt található.

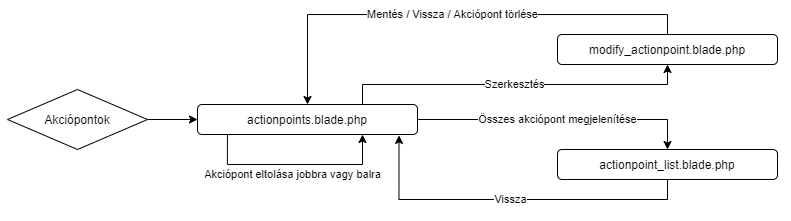


44. ábra: Megbeszéléshez tartozó nézetek

**Actionpoint**

Az alábbi nézetek tartoznak ide a 45-ös ábrán látható módon:

* actionpoints.blade.php: ez a nézet felel az akciópontok oldalért, akciópont módosítás és státusz módosítás után ide kerülünk vissza
* modify\_actionpoint.blade.php: az akcióponton szereplő szerkesztés gombra kattintva kerülünk ide, ez a módosítás nézete
* actionpoint\_list.blade.php: azon megbeszélések akciópontjait jeleníti meg rendezve, amelyeknek tagja a felhasználó



45. ábra: Akcióponthoz tartozó nézetek

**Diary**

A naplóhoz a következő nézetek tartoznak (lásd: 46. ábra):

* diary.blade.php: a felhasználó naplózott adatainak megjelenítéséért felelős nézet
* meeting\_diary.blade.php: a megbeszélésekhez tartozó naplózott adatokat jeleníti meg

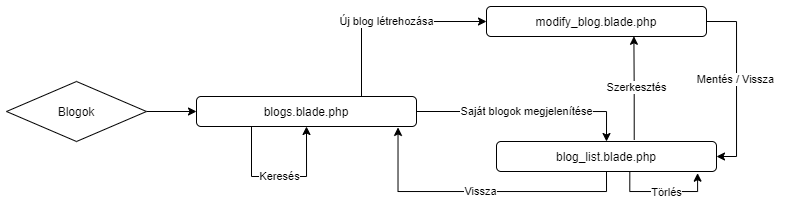


46. ábra: Naplóhoz tartozó nézetek

**Blog**

A blog mappához tartozó nézetek az alábbiak (lásd: 47. ábra):

* blogs.blade.php: a blogbejegyzések megjelenítéséért felelős nézet, keresés után a szűrt blogok itt jelennek meg
* modify\_blog.blade.php: az ide mutató útvonaltól függ, hogy új blog létrehozásának vagy blog módosításának a nézete jelenik meg
* blog\_list.blade.php: a felhasználó saját blogjait jeleníti meg, blog létrehozása, módosítása, törlése után ide kerülünk vissza



47. ábra: Bloghoz tartozó nézetek

## 3.9 Segédfüggvények

Az alkalmazáson belül egyes táblák értékeinek megjelenítéséhez, átlagok kiszámításához segédfüggvények lettek létrehozva, melyek az *app* mappán belül a helpers.php fájlban találhatók meg. A függvények a következők:

* weather\_average(weather\_report): a felhasználó által beküldött értékek alapján visszaadja az átlagra jellemző időjárás-elemet – napos, felhős, esős vagy viharos. Ez az alábbi kódrészleten tekinthető meg:

*//időjárás jelentés átlag*

function **weather\_average**($weather\_report){

    if((($weather\_report['1'] +  $weather\_report['2'] +  $weather\_report['0'])/3) <= 1.5)

        return 'napos';

    elseif ((($weather\_report['1'] +  $weather\_report['2'] +  $weather\_report['0'])/3) <= 2.5)

        return 'felhős';

    elseif ((($weather\_report['1'] +  $weather\_report['2'] +  $weather\_report['0'])/3) <= 3.5)

        return 'esős';

    else

        return 'viharos';

}

* weather\_value(weather\_report, num): mivel az időjárás elemek számként vannak lementve, ezért szükség egy függvényre, mely visszaadja az adott időjárás elemek elnevezését
* weather\_sum\_average(reports): visszaadja a csapat összesített időjárás jelentés átlagát
* get\_status(stat): a Kanban oszlop értékeit adja vissza
* get\_feeling(feel): a plusz-mínusz kártya hangulatát adja vissza
* form\_sum\_average(forms): az űrlap eredmények összesített átlagát adja vissza

*//űrlap összesített átlag*

function **form\_sum\_average**($forms) {

    $forms\_sum = 0;

    $c = **count**($forms);

    for ($i = 0; $i < $c; $i++) {

        for ($j = 0; $j < 5; $j++) {

            $forms\_sum += $forms[$i]->form[$j];

        }

    }

    return ($forms\_sum/($c\*5));

}

* form\_sum\_average\_com(forms): az űrlap kommunikáció sorára vonatkozó összesített átlageredményt adja vissza
* form\_sum\_average\_help(forms): az űrlap egymás segítése, támogatása sorára vonatkozó összesített átlageredményt adja vissza
* form\_sum\_average\_respect(forms): az űrlap tisztelet sorára vonatkozó összesített átlageredményt adja vissza
* form\_sum\_average\_share(forms): az űrlap tehermegosztás sorára vonatkozó összesített átlageredményt adja vissza
* form\_sum\_average\_speed(forms): az űrlap munkavégzés sebessége sorára vonatkozó összesített átlageredményt adja vissza

## 3.10 Bővítési lehetőségek

A MultiRetro webes alkalmazást számtalan módon lehet bővíteni. Az alábbiakban leírom azokat a terveket, amiket szívesen megvalósítottam volna, ha több idő áll rendelkezésre.

**Akciópontok struktúrájának módosítása**

Jelenleg az alkalmazásban egy akciópont csak egy felhasználóhoz tartozhat. Azonban ha átgondoljuk a munkafolyamat menetét, akkor célszerű lenne, ha egy akcióponthoz több felhasználó is tartozhatna, és együtt tudnák szerkeszteni azt.

**Szerepkörök beiktatása**

Mivel az alkalmazás támogatja az agilis szoftverfejlesztési módszertanokat, így célszerű lenne bevezetni különböző szerepköröket. Scrum használatánál pl. Scrum master – Project owner – Developers szerepköröket. Minden szerepkörnek más jogosultságot biztosítana az oldal, pl.: a fejlesztők nem tudnak megbeszélést és csapatot létrehozni.

**Regisztráció kihagyása**

A felhasználók regisztráció nélkül is tudják használni az alkalmazást, így egyszerűsítve a megbeszélésekbe való belépést. Természetesen a blogbejegyzés írását, és a napló megtekintését továbbra is csak regisztrált felhasználóknak tenném elérhetővé.

**Új technikák hozzáadása**

Jelenleg a megbeszélésen 3 db választható technika elérhető: időjárás jelentés, plusz-mínusz tábla és űrlap.

Az egyik új technika lehetne az idősáv beépítése, melyre a felhasználók különböző jeleket helyezhetnek el, mellyel kifejezik, hogy az adott időszakra visszatekintve milyennek látják azt. A jelekhez megjegyzést is lehet rendelni, és természetesen ezek az adatok is bekerülnének a naplóba.

A másik technika lehetne akár a csoportosítás, ahol a weblapon kártyák jelennek meg, és ezeket lehetne rendezni, priorizálni. A kialakult kép ugyanúgy visszatekinthető lenne a naplóból.

Természetesen rengeteg lehetőség és technika áll rendelkezésre az agilis visszatekintők során, azonban akár újakat is ki lehet találni. A lehetőségek tárháza szinte végtelen.

**Grafikus effektek megjelenítése**

Sokkal interaktívabbá tenné a megbeszélést, ha a felhasználó nem gombokkal tudná csak irányítani a különböző kártyák létrehozását, mozgatását, hanem a kurzorral is lehetőség lenne irányítani azt.

## 3.11 Tesztelés

Az alkalmazás tesztelése két részben valósult meg automatikus tesztek és manuális tesztek segítségével. Az agilis kutatócsoport közreműködésével három videó [10] is létrejött, melyek olyan retrospektíveket mutatnak be, amelyek a MultiRetro segítségével valósultak meg. Tehát a MultiRetro alkalmazás élesben is tesztelve lett kétszer.

### 3.11.1 Automatikus tesztek

Az automatikus tesztek a Laravel beépített tesztelőjével hajtódnak végre. A tesztek futtatásához szükséges beírni a terminálba a következő parancssort: *./vendor/bin/phpunit*

Kétféle típusú tesztelési módszer figyelhető meg, melyek két mappába vannak csoportosítva:

* Feature: az alkalmazás alapját - az adatbázis felépítését, a táblák közötti kapcsolatot és a seederek működését - teszteli.
* Unit: az alkalmazás belépés, regisztráció és kilépés funkcióját, valamint a fő útvonalak sikeres elérését teszteli.

**Feature:**

* DatabaseCountTest: seedelés után megvizsgálja, hogy minden adat el lett mentve az adatbázisba oly módon, hogy a táblák sor összege megegyezik-e a megszámlálás függvény által visszaadott értékkel.
  + test\_have\_7\_users: az alább látható kódrészlet mutatja be

*//felhasználók*

    public function **test\_have\_7\_users**()

    {

        $this->**assertEquals**(7, **User**::**count**());

    }

* + test\_have\_4\_teams
  + test\_have\_5\_meetings
  + test\_have\_10\_actionpoints
  + test\_have\_7\_plus\_minus\_tasks
  + test\_have\_5\_blogs
  + test\_have\_5\_diaries
* DatabaseRelationshipTest: seedelés után a modellekben szereplő kapcsolatok segítségével teszteli le, hogy az adatbázis kapcsolatrendszere megfelelően működik-e.
  + test\_user\_team\_relationship: a következő kódrészlet mutatja be:

*//felhasználó - csapat*

    public function **test\_user\_team\_relationship**()

    {

        $user = **User**::**find**(1);

        $teams = $user->owner\_teams;

        $team = $user->owner\_teams->**find**(1);

        $this->**assertEquals**(2, $teams->**count**());

        $this->**assertEquals**($user->id, $team->team\_owner->id);

    }

* + test\_user\_meeting\_relationship
  + test\_user\_actionpoint\_relationship
  + test\_user\_plus\_minus\_task\_relationship
  + test\_user\_blog\_relationship
  + test\_user\_diary\_relationship
  + test\_meeting\_diary\_relationship
  + test\_meeting\_actionpoint\_relationship
  + test\_meeting\_plus\_minus\_task\_relationship
  + test\_team\_meeting\_relationship
  + test\_team\_user\_relationship

**Unit:**

* AuthTest: a belépés, regisztráció és kijelentkezés funkciókat teszteli jó és rossz bemeneti adatok segítségével
  + test\_user\_can\_see\_login
  + test\_good\_login
  + test\_bad\_login
  + test\_user\_redirect\_to\_home\_after\_login
  + test\_user\_can\_logout
  + test\_user\_can\_see\_registration: az alábbi kódrészlet mutatja be:

    public function **test\_user\_can\_see\_registration**()

    {

        $response = $this->**get**('/register');

        $response->**assertSuccessful**();

        $response->**assertViewIs**('auth.register');

    }

* + test\_good\_register: az alábbi kódrészlet mutatja be:

    public function **test\_good\_register**()

    {

        $response = $this->**post**('/register', [

            'name' => 'Test User',

            'email' => 'test@test.com',

            'password' => 'asdfasdf',

            'password\_confirmation' => 'asdfasdf',

        ]);

        $response->**assertRedirect**('/home');

        $users = **User**::**all**();

        $this->**assertAuthenticatedAs**($user = $users->**last**());

        $this->**assertEquals**('Test User', $user->name);

        $this->**assertEquals**('test@test.com', $user->email);

        $this->**assertTrue**(**Hash**::**check**('asdfasdf', $user->password));

    }

* + test\_bad\_register
* RouteTest: teszteli, hogy a fő útvonalak megjelennek-e, vagyis a ’GET’ kérés után 200-as státuszkódot kapunk-e vissza.
  + test\_userprofile\_view: az alábbi kód mutatja be:

*//felhasználó nézet*

    public function **test\_userprofile\_view**() {

        $this->**be**(**User**::**find**(1));

        $response = $this->**get**('/userprofile');

        $response->**assertStatus**(200);

    }

* + test\_teams\_view
  + test\_team\_list\_view
  + test\_meetings\_view
  + test\_meeting\_list\_view
  + test\_actionpoints\_view
  + test\_actionpoint\_list\_view
  + test\_blogs\_view
  + test\_blog\_list\_view
  + test\_diary\_view

### 3.11.2 Manuális tesztek

A manuális tesztek elsősorban a felület működését ellenőrzik, valamint lefedik azokat a részeket, amik az automatikus tesztek során nem lettek ellenőrizve.

Az alábbiakban képernyőnként elkülönítve találhatók meg a tesztek leírással együtt:

**Kezdőlap:**

* *Bejelentkezés:* sikeres belépés után megjelenik a navigációs sáv az oldal tetején: Profil, Csapatok, Megbeszélés, Akciópontok, Blogok, Napló, Kijelentkezés, és a Profil képernyő jelenik meg.
* *Regisztráció:* sikeres regisztráció után megjelenik a navigációs sáv, és a Profil képernyő jelenik meg.

**Profil:**

* *Megjelenés:* A felhasználó adatai megjelennek a képernyőn, alattuk látható az ’Adatok módosítása’ gomb.
* *Adatok módosítása:* A gombra kattintva megjelenik az adatok módosítása nézet
  + *Eredeti adatok betöltése:* A kettő bemeneti mezőben – Felhasználó neve, E-mail cím - megjelennek az eredeti adatok.
  + *Felhasználó név validációja:* Ha üres vagy túl hosszú nevet adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *E-mail cím validációja:* Ha rossz formátumban adjuk meg az e-mail címet, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Sikeres mentés:* A ’Mentés’ gombra kattintva frissülnek a felhasználó adatai, a ’Profil’ nézetre visszairányítva láthatóak a frissített adatok.
  + *Vissza gomb:* A vissza gombra kattintva a Profil nézetre kerülünk vissza.

**Csapatok:**

* *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt, az oldal tetején elérhető az ’Új csapat létrehozása’ és az ’Összes csapat listázása’ gomb, valamint a saját csapatok alatt látható a ’Szerkesztés’ és ’Törlés’ gomb.
* *Új csapat létrehozása, szerkesztés:* 
  + *Eredeti adatok betöltése:* Ha a ’Szerkesztés’ gombra kattintva jutottunk ide, akkor a csapat neve megjelenik, és a választható csapattagok közül ki vannak pipálva az eredeti tagok.
  + *Csapat nevének validációja:* Ha üres vagy túl hosszú nevet adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Választható csapattagok validációja:* Ha nem jelölünk ki egy csapattagot sem, akkor a ’Mentés’ gombra kattintva visszatérünk a szerkesztéshez a megfelelő hibaüzenettel.
  + *Vissza gomb:* A vissza gombra kattintva a Csapatok nézetre kerülünk vissza.
* *Felhasználó által létrehozott csapat törlése:* A csapat alatt található ’Törlés’ gomb csak akkor jelenik meg, ha az adott felhasználó hozta létre a csapatot. Erre kattintva felugrik egy figyelmeztetés: „Biztosan törölni szeretné a csapatot? Minden hozzá tartozó megbeszélés és naplózott adat törlődni fog!”. Ha igen-re kattintunk, akkor a csapat, és minden vele kapcsolatban lévő adat – megbeszélések, akciópontok, plusz-mínusz-kártyák, naplózott adatok – törlődnek.
* *Összes csapat listázása:* Az ’Összes csapat listázása’ gombra kattintva listába rendezve megjelennek a csapatok.
  + *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt rendezve – csapat neve, létrehozója, tagok.
  + *Vissza gomb:* A vissza gombra kattintva a Csapatok nézetre kerülünk vissza.

**Megbeszélések:**

* *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt, vagyis csak azok a megbeszélések, amiket a felhasználó hozott létre, vagy tagja, továbbá az oldal tetején elérhető az ’Új megbeszélés létrehozása’ és az ’Összes megbeszélés listázása’ gomb, valamint a saját megbeszélések alatt látható a ’Szerkesztés’ és ’Törlés’ gomb.
* *Új megbeszélés létrehozása, szerkesztés:* 
  + *Eredeti adatok betöltése:* Ha a ’Szerkesztés’ gombra kattintva jutottunk ide, akkor a megbeszélés neve és a dátum megjelenik, valamint a választható csapatok közül ki van jelölve az eredeti, és a választható technikák közül ki vannak pipálva az eredetiek.
  + *Megbeszélés nevének validációja:* Ha üres vagy túl hosszú nevet adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Dátum validációja:* Ha az aktuális napnál régebbi időpontot adunk meg vagy nem töltjük ki a mezőt, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Választható csapat validációja:* Ha nem jelölünk ki egy csapatot sem, akkor a ’Mentés’ gombra kattintva visszatérünk a szerkesztéshez a megfelelő hibaüzenettel.
  + *Választható technikák validációja:* Nincs beépített hibaüzenet, ha nem jelölünk be egyetlen technikát sem, mivel a technikák használata nem kötelező.
  + *Vissza gomb:* A vissza gombra kattintva a Megbeszélések nézetre kerülünk vissza.
* *Felhasználó által létrehozott megbeszélés törlése:* A megbeszélés alatt található ’Törlés’ gomb csak akkor jelenik meg, ha az adott felhasználó hozta létre a megbeszélést. Erre kattintva felugrik egy figyelmeztetés: „Biztosan törölni szeretné a megbeszélést? Minden hozzá tartozó akciópont és naplózott adat törlődni fog!”. Ha igen-re kattintunk, akkor a megbeszélés, és minden vele kapcsolatban lévő adat – akciópontok, plusz-mínusz-kártyák, naplózott adatok – törlődnek.
* *Összes megbeszélés listázása:* Az ’Összes megbeszélés listázása’ gombra kattintva listába rendezve megjelennek a még aktív megbeszélések.
  + *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt rendezve – megbeszélés neve, létrehozója, dátum, csapat.
  + *Vissza gomb:* A vissza gombra kattintva a Megbeszélések nézetre kerülünk vissza.
* *Megbeszélés megnyitása:* A megbeszélések neve mellett található ’Megnyitás’ gombra kattintva megjelenik az adott megbeszélés felülete.
  + *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt. Elsőként a megbeszélés adatai – létrehozója, dátum, csapat neve, csapattagok, majd azok a technikák, amik ki lettek választva létrehozáskor, a következő sorrendben: időjárás jelentés, plusz-mínusz kártyák és az űrlap. A lap alján szerepel az akciópont létrehozása és a Kanban tábla a megbeszéléshez tartozó akciópontokkal együtt. Legalul a ’Megbeszélés lezárása’ gomb található.
  + *Időjárás jelentés megjelenése:* A táblázat 5 oszlopos és 4 soros, a fejlécben megtekinthetők az időjárás képek – napos, felhős, esős, viharos, valamint oldalt olvasható a Teljesítmény, Együttműködés és Közérzet felirat.
  + *Időjárás jelentés adatainak küldése:* Nincsen kötelezően kitöltendő mező, mivel a felhasználóra van bízva mire szeretne választ adni. Az ’Időjárás jelentés küldése’ gombra kattintva a naplóba lementődnek az adatok, és a gomb helyett az eredmények jelennek meg. Ez tartalmazza a felhasználó időjárásra adott saját átlagát és a csapat együttes átlagát. Az átlag helyesen számolódik ki.
  + *Plusz-mínusz kártya leírásának validációja:* Ha üres vagy túl hosszú leírást adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Plusz-mínusz kártya tulajdonságának validációja:* Ha nem adjuk meg a kártya típusát, akkor a ’Mentés’ gombra kattintva visszatérünk a szerkesztéshez a megfelelő hibaüzenettel.
  + *Plusz-mínusz tábla megjelenése:* A tábla két oszlopos – Pozitív, Negatív – és a kártyák egymás alatt vannak felsorakozva. A pozitív oldalon piros, a negatívon kék színűek a kártyák, és a bal felső sarokban egy piros plusz jel, míg a jobb felső sarokban egy kék mínusz jel található.
  + *Plusz- és mínusz jelek működése:* A plusz jelre kattintva eggyel nő a kártya bal felső sarkában látható szám értéke, míg a negatív jelre kattintva eggyel nő a kártya jobb felső sarkában található szám értéke. Saját kártyát nem lehet értékelni.
  + *Űrlap megjelenése:* A táblázat 4 oszlopos és 6 soros, a fejlécben megtalálható az 1, 2, 3, 4, 5 felirat, míg oldalt a Kommunikáció, Egymás segítése támogatása, Tisztelet, Tehermegosztás és Munkavégzés sebessége felirat.
  + *Űrlap adatainak küldése:* Az űrlap kitöltése nem kötelező, így nincs hozzá tartozó validáció. Az ’Űrlap beküldése’ gombra kattintva az adatok lementődnek a naplóba, majd a táblázat helyén egy felirat jelenik meg: Az űrlap sikeresen el lett mentve! Alatta megjelenik a csapat együttes átlaga és külön-külön a készségek átlaga.
  + *Akciópont leírásának validációja:* Ha üres vagy túl hosszú leírást adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Akciópont választható felhasználók validációja:* Ha nem adjuk meg a hozzá tartozó felhasználót, akkor a ’Mentés’ gombra kattintva visszatérünk a szerkesztéshez a megfelelő hibaüzenettel.
  + *Kanban tábla megjelenése:* A tábla 3 oszlopos, a fejlécben a következő feliratok találhatók: Megvalósításra vár, Megvalósítása folyamatban, Megvalósítva. A megfelelő oszlop alatt felsorakoznak az akciópontok. Az első oszlopnál piros színűek és a jobb felső sarokban található egy nyíl, a középső oszlopban zöldek, és a bal és jobb felső sarokban található nyíl, míg a harmadik oszlopban a bal felső sarokban található nyíl.
  + *Akcióponton található nyilak működése:* A jobb fele mutató nyílra kattintva az akciókártya egy oszloppal jobbra csúszik, míg a bal nyílra kattintva egy oszloppal balra csúszik. A rendszer koherens, mivel az Akciópontok menüpont alatt is látható a változás.
  + *Megbeszélés lezárása:* A ’Megbeszélés lezárása’ gombra kattintva a következő üzenet ugrik fel: „Biztosan le szeretné zárni a megbeszélést?” Az igen-re kattintva visszakerülünk a Megbeszélések oldalra, és nem látjuk a lezárt megbeszélést az oldalon, azonban a Napló menüpont alatt meg fog jelenni.
  + *Vissza gomb:* A vissza gombra kattintva a Megbeszélések nézetre kerülünk vissza. Látható az előbb megnyitott megbeszélés, és újra megnyitva azt az adatok nem tűnnek el, onnan lehet folytatni a kitöltést, ahol abba lett hagyva.

**Akciópontok:**

* *Megjelenés:* A megfelelő adatok jelennek meg egy Kanban táblába rendezve. Az akciókártyák alján megtalálható a ’Szerkesztés’ gomb, és a lap tetején az ’Összes akciópont megjelenítése’ gomb.
* *Akcióponton található nyilak működése:* Az első oszlop kártyáin csak a jobboldali nyíl, a második oszlop kártyáin mind a két nyíl, a harmadik oszlop kártyáin csak a bal nyíl jelenik meg. A jobb fele mutató nyílra kattintva az akciókártya egy oszloppal jobbra csúszik, míg a bal nyílra kattintva egy oszloppal balra csúszik.
* *Összes akciópont megjelenítése:* Az ’Összes akciópont megjelenítése’ gombra kattintva megtekinthetővé válnak megbeszélés szerint csoportosítva és Kanban táblába rendezve az akciópontok. Azonban csak azokat a megbeszéléseket láthatja a felhasználó, aminek tagja.
  + *Vissza gomb:* A vissza gombra kattintva az Akciópontok nézetre kerülünk vissza.
* *Akciópont szerkesztése:* az akcióponton található ’Szerkesztés’ gombra kattintva megjelenik az akciópont szerkesztése lehetőség.
  + *Eredeti adatok betöltése:* Az akciópont leírása megjelenik, és a hozzá tartozó felhasználó van megjelölve.
  + *Akciópont leírásának validációja:* Ha üres vagy túl hosszú leírást adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + Akciópont törlése: Az ’Akciópont törlése’ gombra kattintva felugrik a következő üzenet: „Biztosan törli az akciópontot?” Az igen-re kattintva az akciópont törlődik, és visszakerül a felhasználó az Akciópontok nézetre.
  + *Vissza gomb:* A vissza gombra kattintva az Akciópontok nézetre kerülünk vissza.

**Blogok:**

* *Megjelenés:* A megfelelő adatok jelennek meg egymás alatt, az oldal tetején elérhető az ’Új blog létrehozása’ és az ’Saját blogok listázása’ gomb, valamint a gombok alatt található a keresés.
* *Új blog létrehozása, szerkesztése:*
  + *Eredeti adatok betöltése:* A ’Szerkesztés’ gomb csak az adott felhasználó által létrehozott blogok alatt jelenik meg, ha erre a gombra kattintva jutottunk ide, akkor a blog címe, leírása és a címkék megjelennek.
  + *Blog címének validációja:* Ha üres vagy túl hosszú címet adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Blog leírásának validációja:* Ha üres leírást adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet. A blog hossza nincs korlátozva.
  + *Első címke validációja:* Ha üres vagy túl hosszú címkét adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet.
  + *Második címke validációja:* Ha túl hosszú címkét adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet. Kitöltése nem kötelező.
  + *Harmadik címke validációja:* Ha túl hosszú címkét adunk meg, akkor a ’Mentés’ gombra kattintva megjelenik a megfelelő hibaüzenet. Kitöltése nem kötelező.
  + *Vissza gomb:* A vissza gombra kattintva a Saját blogok nézetre kerülünk vissza.
* *Saját blogok megjelenítése:* A ’Saját blogok megjelenítése’ gombra kattintva egymás alatt megjelennek a felhasználó által írt blogok. Minden blog alatt megtalálható a ’Szerkesztés’ és ’Törlés’ gomb.
  + *Blog törlése:* A ’Törlés’ gombra kattintva felugrik a következő szöveg: „Biztosan törli a blogot?” Az igen-re kattintva törlődik a blog.
  + *Vissza gomb:* A vissza gombra kattintva a Megbeszélések nézetre kerülünk vissza.
* *Keresés a blogok között:* A megfelelő szót beírva és a ’Keresés’ gombra kattintva csak azok a blogok jelennek meg, amiknek valamelyik címkéje a keresett szó.

**Napló:**

* *Megjelenés:* A felhasználó naplózott adatai jelennek meg megbeszélés szerint. Legfelül azok a megbeszélések jelennek meg, amik nem aktívak, és a felhasználó részt vett rajtuk.
* *Megbeszélés megnyitása:* A megbeszéléshez tartozó naplózott akciópontok és plusz-mínusz kártyák jelennek meg, valamint legfelül az összessített átlagok láthatók – időjárás jelentés és űrlap.
* *Vissza gomb:* A vissza gombra kattintva a Napló nézetre kerülünk vissza.

A fentebb kifejtett tesztek az alábbi eszközökön lettek sikeresen futtatva:

* Xiaomi Mi A1 okostelefon (Android 9)
* Xiaomi Mi 9 Lite (Android 10)
* Asztali számítógép: processzor: Intel Core i3-4330 3.50 GHz; memória: 8 GB; 64 bites operációs rendszer
* Lenovo Ideapad 520: processzor: Intel Core i5-8250U 1.6 GHz; memória: 8 GB; 64 bites operációs rendszer
* Samsung Galaxy S8 (Android 9)

# 4. Forrásjegyzet

1. „Laravel Documentation” [Online]. Link: <https://laravel.com/docs/8.x>

Elérés dátuma: 2021. 02. 10.

1. „Bootstrap Documentation” [Online]. Link: <https://getbootstrap.com/docs/5.0/getting-started>

Elérés dátuma: 2021. 02. 12.

1. „SQLite Documentation” [Online]. Link: <https://www.sqlite.org/docs.html>

Elérés dátuma: 2021.02.27.

1. „Tóta Dávid github repository - szerveroldali webprogramozás” [Online]. Link: <https://github.com/totadavid95/PhpComposerInstaller>

Elérés dátuma: 2021. 03. 21.

1. Bhuvan Unhelkar: Software Engineering with UML, CRC Press - Taylor & Francis Group, 2018, [427], ISBN13- 978-1-138-29743-2
2. „Scrum retrospective” [Online]. Link: <https://www.scrum-tips.com/agile/scrum-agile/scrum-meetings/sprint-retrospective/scrum-retrospective-4-decide-what-to-do/>

Elérés dátuma: 2021.04.12.

1. Esther Derby, Diana Larsen: Agile Retrospectives: Making Good Teams Great, Pragmatic Bookshelf, 2006, [170], ISBN-0977616649
2. „Heroku: Cloud Application Platform” [Online]. Link: <https://devcenter.heroku.com/categories/reference>

Elérés dátuma: 2021.04.29.

1. „GitHub elérhetőség” [Online]. Link: <https://github.com/baloghemi/szakdoga_repo>

Elérés dátuma: 2021.04.14.

1. „Videó a tesztelésről” [Online]. Link: <https://ikelte-my.sharepoint.com/:f:/g/personal/ii1ank_inf_elte_hu/Ei3GGXyS4BNEoS0AERjCVDoBc_l_2uuBeZ0Itra55SXZtQ?e=4bKuRj>

Elérés dátuma: 2021.05.06.

1. Agile Manifesto: [https://agilemanifesto.org/iso/hu/manifesto.html](%20https:/agilemanifesto.org/iso/hu/manifesto.html) [↑](#footnote-ref-1)
2. Scrum Guide: <https://scrumguides.org/> [↑](#footnote-ref-2)
3. Installation: <https://laravel.com/docs/8.x/installation>

   Deployment: <https://laravel.com/docs/8.x/deployment> [↑](#footnote-ref-3)
4. <https://drawio-app.com/tutorials/> [↑](#footnote-ref-4)
5. <https://www.mockflow.com/> [↑](#footnote-ref-5)
6. <https://www.gimp.org/> [↑](#footnote-ref-6)
7. DB Browser for SQLite: <https://sqlitebrowser.org/> [↑](#footnote-ref-7)