LIVE IN CARE

szoftver

Szeged, 2020

Szegedi Szakképzési Centrum Vasvári Pál Gazdasági és Informatikai Szakgimnáziuma
Az 54 213 05 számú Szoftverfejlesztő szakképesítés záródolgozata

Bálint István
Istvann.balint@gmail.com

1. Tartalomjegyzék:

Tartalom

1. Tartalomjegyzék: 1
2. Bevezetés: 3
3. Fejlesztői dokumentáció: 4
   I. Adatbázis 4
      a. Egyedek: 4
      b. Belső tulajdonságok: 5
      c. Egyedek, tulajdonságok típussal és névvel: 5
      d. Külső tulajdonságok: 7
      e. Összekapcsolásuk: 7
   II. Szoftver 8
      a. Verziókövetés 9
      b. Felhasználók: 9
      c. Gépigény: 10
      d. Interjúk: 10
      e. Tervezés 11
      f. Bejelentkezés: 12
      g. Felépítése, mappa szerkezet 13
      h. Hozzáadás, módosítás, törlés: 13
      i. Unit teszt 20
      j. Hibakezelés: 20
      k. Összefüggés a weblappal 21
      l. Fejlesztési lehetőség 22
   III. A webalkalmazás 22
      a. Bevezetés, probléma felvetés 22
      b. Felhasználók 23
      c. Csatlakozás adatbázishoz 23
      d. Homepage (kezdő lap): 23
      e. Regisztrációs felhasználónév és jelszó használata: 25
      f. Adatain ellenőrzése és jelszó változtatás: 26
      g. Fájlok, szerkezeti felépítés 28
      h. Reszponzivitás 28

i. Fejlesztési lehetőség: 29
4. Felhasználói dokumentáió: 29
I. Szoftver 29
a. Telepítés 29
b. Bejelentkezés: 30
c. Felvétel, módosítás, törlés, keresés 30
d. Pozíció szerinti felülettek: 31
e. Hiba jelzés 34
II. Weblap 34
a. Kezdő lépernyő részei: 34
b. Bejelentkezés 34
c. Adatok ellenőrzése regisztráció után: 35
d. Gyerek követés, Ellátás menü pontok 35
5. Ergonómia 37
6. Összegzés 37
7. Köszönet nyilvánítás 37
8. Plágium nyilatkozat 38
9. Mellékletek: 39
I. E-K diagramm (I. ábra) 39
II. Bachmann-ábra (II. ábra) 39
III. Use-Case diagramm (III. ábra) 40

2. Bevezetés:

A záródolgozatom egy olyan szoftvert hivatott megvalósítani, ami az állami nevelésben élő gyerekek nevelkedését segíti elő. Valamint az adott intézményben a dolgozók munkáját teszi gördülékenyebbé. Emellett megkönnyíti a rendszerben lévő felhasználók (lásd: Fejlesztői dokumentáció -> Felhasználók) információ áramlását.

Fontos emellett megemlíteni a más problémákat is. Papír alapon dolgoznak nagy részt az intézmények ilyen szférában különböző dokumentáció elkészítésére. Ezért ez a szoftver fel tudja váltani a papírt, hiszen így elektronikusan tudjuk tárolni a személyes adatkat. Ez sok időt vesz igénybe az egyes nyomtatványok kitöltése, és ezeket iktatni is kell. Kitöltéssel is lehetnek problémák, amit például az Intézmény ügyintéző nem tud kiolvasni, így vagy rossz vagy hiányos információ lesz a gyerekről. Problémát okoz, akár egy dokumentum elő keresése, mert azt egy nagy irattartó mappában tartják. Természetesem az is előfordult már, hogy nem lett meg az adott dokumentum, mert elveszett. Egyik célja, hogy modernizáljuk az intézmény telekommunikációs infrastruktúráját (felek közti gyors információ átadását), és a gyors adat elérést (gyermek személyes adatai). Ha csak egy státusz változás[1] veszünk figyelembe, az több hónapot is igénybe vehet. Sajnos a fiatalnak várnia kell. Több problémát is felvett ez, például a gyerek nem tudja gyorsan múltat lezárni, vagy ha fontos a gyors áthelyezés egyéb okok miatt[2].

Úgy gondolom ahhoz, hogy a gyerek fejlesztése szóba jöjjön, ahhoz természetesen mindent tudni kell a gondozott fiatalról. Hiszen anélkül nem tudhatjuk, milyen állapotban van a gyerek. Milyen lelki vagy fizikai sérülése van, ha van. Hiszen a szülőktől való leválasztás folyamata[3] igen nehéz tud lenni minden fiatal számára. Ezért is kulcs fontosságú a fiatal nyomon követése és szoros együtt működése a szülőkkel. Ezzel a program csomaggal őket is szeretnék támogatni, informálni a saját gyerekéről.

Tapasztalataim alapján is mondhatom azt, hogy nem feltétlen figyelnek oda a gyerek egyes nehézségeire (SNI, BTM) egyes intézményekben. Hanem átlagos gyereként kezelik. A program által a nevelő oda tud figyelni jobban a gyerekre. Mert amit a Pszichológus tanácsol egy vizsgálat során azt azonnal fel tudja tölteni az adatbázisba és a nevelő azonnal informálva lesz, így a gyerek kis gondjait szem ellőtt tudja tartani. De ez más vizsgálatoknál is kézen fekvő.

Az előbbi okok folytán határoztam el, hogy ezt a munkát segíteni szeretném egy webes és egy desktop felülettel. Szerepkörök alapján lesz a beléptetés, minden szerepkör egy adott részhez nyúlhat / módosíthat. Ez felhasználó jogoktól és munka pozíciótól függnek. Nagy részt adatok tárolása lesz a célja. De amint említettem a nevelőt segíteni fogja, ha meg tudja időben, hogy milyen problémái lehetnek a gyereknek, hiszen a Pszichológus szakvéleményt ír. Fontos része

[1] Ennek több fajtája is van. Pl.: utógondozottság (ha fiatal 18 élet évét betöltötte, akkor átkerülhet egy olyan intézménybe, ahol csak 18 év feletti fiatalok vannak.)

[2] Akár a nevelő bántalmazza a gyereket vagy / mert nem tud a gyerek és a nevelő együtt működni.

[3] Az, amikor a gyerek frissen kerül az intézménybe és elfogadtatni vele, hogy most már itt fog lakni, nem a szüleivel, de szülök.

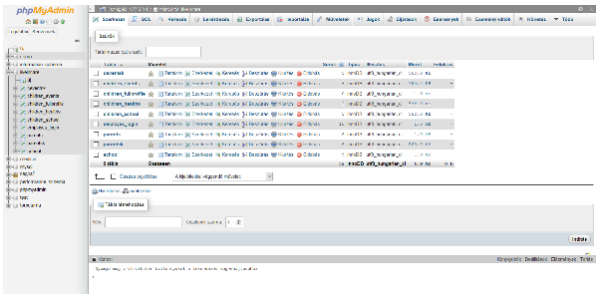
lesz a szoftvernek, hogy a szülök is meg tudjanak a gyerekekről információkat, hogy merre járt, milyen sport eredményeket ért el, milyen egészségügyi vizsgálatokon vett részt. Hiszen a szülő elsődleges célja alap esettbe, hogy tudja magától nevelni a gyereket, vagyis kihozza a fiatalt az intézmény falai közül, ehhez egy alapot kell megteremtenie, ami sok munkával járhat. Fontos megejteni, hogy ez is beléptetés alapján lesz, és csak is akkor léphet be a szülő, ha ehhez a gyerek 100%-osan hozzá járult, engedélyhez kötött. Természtesen az átláthatóság is szerepet játszik a szoftver létrehozásában. Jobban meg tudják vizsgálni melyik fiatal pontosan melyik intézményben tartozkodik (melyik területi intézményben).

3. Fejlesztői dokumentáció:

Egy szoftver megtervezése során fontos, szempont a programtervezés folyamata, ami több részből tevődik ki. Adottság, lehetőség, korlát, elemzés, szerződés, részletes terv, kódolás, tesztelés, program bevezetés és átadás. Nekünk most első négy szempont és a részletes terv, valamint a kódolás, amik a fontosak. Felkel mérnünk, hogy milyen adottsággal rendelkezi az adott intézmény. Milyen lehetőségek vannak (pl.: több intézmény van egy helyen), és az egyik legfontosabb milyen korlátok vannak (pl.: puszta közepén van az intézmény, rossz internet kapcsolat). Elemzés, hogy milyen a jelenlegi rendszer adottságai (fejlesztés lehetőség). Részletes terv pedig a leírása annak, mi szükséges az adott szoftverhez.

I. Adatbázis

Adatbázis szerves részét képezi egy szoftvernek, hiszen itt tárlódnak az adatok és innen lehet lekérdezni az adatokat. A tanulmányaim során a phpMyAdmin (MySQL) adatbázis kezelő felületet használtam, ennek okén fogva a szakdolgozatomba is ezt alkalmazom. Szükségem volt egy virtuális saját szervere és ehhez kellett a XAMPP nevezetű program (*0. ábra*). Virtuális szerveren „találkozik” a weblap és az asztali alkalmazásunk. Fontos előre megadni az adatbázisunk irány vonalait. És azzal kezdeni. Én is így terveztem meg szoftver és a weblapomat. Nyilván ebben a későbbiekben változtattam, ahogy a funkciókat programoztam. Adatbázis biztonság is fontos főleg a mai modern világban. Az adatbázis csak szigorúan, az asztali és weblap által elérhető. Külön adatbázisban vannak a táblák, más nem használja ezt az adatbézis.



0. ábra: phpMyAmin/XAMPP

a. Egyedek:

Az egyed az, amit le akarunk írni, amelynek az adatait tároljuk és gyűjtjük az adatbázisban. Az egyedet idegen szóval entitásnak nevezzük. Egyednek tekinthetünk például egy személyt. Fontos meghatározni milyen adatokra van szükségünk, nekem ezek az interjúk alatt derültek ki, mi az, ami nagyon fontos adat a gyerekről, vagy más egyedekről.

- Gyerekek teljes profilja (**children_fullprofile**)
- Szülők tábla, ahova felregisztrálják őket (**parents**)

- Gyerek egészségügyi vizsgálatai és pszichológiai vizsgálata eredményét itt tároljuk (**children_healths**)
- Gyerekkel történt események szülőknek, gyerek hozzárendelés nélkül (**children_events**)
- Iskolai intézmények legfontosabb adatai ide regisztrálhatjuk fel. (**school**)
- Dolgozok adatai és belepésének adatai (**employers_login**)
- Kapcsoló táblák
    - Esemény és a gyerek idja alapján kapcsolja össze, vagyis melyik gyerek milyen eseményen vett részt (**ceventsk**)
    - A gyerekek összekapcsolása szülőkkel, vagyis melyik gyereknek kik a szülei (**parentsk**)
    - Gyerek adott oktatási jogviszonya egy intézménnyel (**children_school)**

b. Belső tulajdonságok:

Egy adatmodellben valamennyi egyedet egyenként véges számú tulajdonsággal írunk le. Ezek a tulajdonságok együttesen alkotják az egyed belső szerkezetét, és az egyed belső tulajdonságainak hívjuk.

- **children_fullprofile** (ID, cname, csex, cidcardnumber, ctajnumber, cbirth, cbirthplace, ccoming, clocation)
- **parents** (ID, pname, pbirth, psex, pidcardnumber, loginpermission, loginuser , loginpsw, reg)
- **children_events** (ID, title, details, img, by)
- **children_healths** (ID, childrenID, type, details, special_treatment,treatdate, by)
- **children_school** (ID, schoolID, childrenID, fromDate, expectedFinish,type, headteacher)
- **employes_login** (ID, ename, emaidenname, esex, ebirth, ebirthplace, ejob, elocation, idcard, euname, epassword)
- **ceventsk** (ID, childrenID, eventsID, timer)
- **parentsk** (ID, pID, childrenID)
- **school** (ID, schoolname, schoolLocation, schoolPhone)

c. Egyedek, tulajdonságok típussal és névvel:

- **children_fullprofile (gyerek tábla):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), gyerek azonosító
    - cname (szöveg, 50 hosszú karaktersorozat), gyerek neve
    - csex (szöveg, 10 hosszú karaktersorozat), gyerek neme
    - cidcardnumber (szöveg, 10 hosszú karaktersorozat), gyerek személyigazolványszáma
    - tajnumber (szöveg, 10 hosszú karaktersorozat), gyerek TAJ száma
    - cbirth (dátum), születési ideje
    - cbirthplace (szöveg, 40 hosszú karaktersorozat), születési helye
    - ccoming (dátum), intézménybe belépés ideje
    - clocation (szöveg, 40 hosszú karaktersorozat), intézmény helye, ahol lakik
- **parents (szülő tábla):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), szülő azonosító
    - pname (szöveg, 40 hosszú karaktersorozat), szülő neve
    - pbirth (dátum), szülő születési ideje
    - psex (szöveg, 10 hosszú karaktersorozat), szülő neme

    - pidcardnumber (szöveg, 10 hosszú karaktersorozat), szülő személyigazolványszám
    - loginpermission (1 bites egész szám), a regisztrációhoz engedély
    - loginuser (szöveg, 25 hosszú karaktersorozat), regisztrációs felhasználó
    - loginpsw (szöveg, 25 hosszú karaktersorozat), regisztrációs jelszó
    - reg (1 bites egész szám), regisztrálást ellenőrző (1 = volt)
- **children_events (esemény tábla):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), esemény azonosító
    - title (szöveg, 30 hosszú karaktersorozat), az esemény címe
    - details (szöveg, 120 hosszú karaktersorozat), az esemény leírása
    - img (kép), kép eseményről
    - by (szöveg, 20 hosszú karaktersorozat), felvevő ember
- **children_healths (vizsgálati tábla):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), vizsgálat azonosító
    - childrenID (egész szám, maximum 11 hosszú, idegen kulcs), gyerek azonosító
    - type (szöveg, 30 hosszú karaktersorozat), vizsgálat típusa
    - details (szöveg, 120 hosszú karaktersorozat), vizsgálat leírása
    - special_treatment (szöveg, 50 hosszú karaktersorozat), különleges bánásmód
    - treatdate (dátum), vizsgálat ideje
    - by (szöveg, 30 hosszú karaktersorozat), orvos neve
- **employes_login (dolgozok táblája):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), vizsgálat azonosító
    - ename (szöveg, 50 hosszú karaktersorozat), dolgozó neve
    - emaidenname (szöveg, 50 hosszú karaktersorozat), dolgozó lánykori neve
    - esex (szöveg, 10 hosszú karaktersorozat), dolgozó neme
    - idcard (szöveg, 10 hosszú karaktersorozat), személyigazolványszáma
    - ebirth (dátum), születési idő
    - ebirthplace (szöveg, 40 hosszú karaktersorozat), születési hely
    - ejob (szöveg, 18 hosszú karaktersorozat), beosztás bejelentkezéshez
    - elocation (szöveg, 25 hosszú karaktersorozat), munka helyszíne
    - euname (szöveg, 25 hosszú karaktersorozat), felhasználó név
    - epasword (szöveg, 25 hosszú karaktersorozat), jelszó

- **children_school (gyerek, amelyik iskolába jár):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), gyerek oktatási jog azonosító
    - schoolID (egész szám, maximum 11 hosszú, idegen kulcs), iskola azonosító
    - childrenID (egész szám, maximum 11 hosszú, idegen kulcs), gyerek azonosító
    - fromDate (dátum), iskola kezdés időpontja
    - exceptedFinish (dátum), iskola várható befejezés időpontja / befejezett suli időpont
    - type (szöveg, 25 hosszú karaktersorozat), típusa
    - headteacher (szöveg, 50 hosszú karaktersorozat), osztályfőnök neve
- **school (oktatási intézmények):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), oktatási intézmény azonosító
    - schoolName (szöveg, 80 hosszú karaktersorozat), oktatási intézmény neve

    - schoolLocation (szöveg, 40 hosszú karaktersorozat), oktatási intézmény helyszíne
    - schoolPhone (szöveg, 14 hosszú karaktersorozat), oktatási intézmény telefonszám
- **ceventsk (esemény gyerekhez kapcsolás):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), esemény- gyerek azonosító
    - childrenID (egész szám, maximum 11 hosszú, idegen kulcs), gyerek azonosító
    - eventsID (egész szám, maximum 11 hosszú, idegen kulcs), esemény azonosító
    - timer (dátum), esemény idópontja

- **parentsk (gyerek szülőhöz kötése):**
    - **ID** (egész szám, maximum 11 hosszú, elsődleges kulcs), esemény- gyerek azonosító
    - pID (egész szám, maximum 11 hosszú, idegen kulcs), szülő azonosító
    - childrenID (egész szám, maximum 11 hosszú, idegen kulcs), gyerek azonosító

d. Külső tulajdonságok:

Az egyedek kapcsolatainak összességét külső tulajdonságoknak hívjuk**.**

- Az egyedek kapcsolatait egy nagyon egyszerű ábrával szemléltethetjük (*I. ábra*). Itt láthatjuk, milyen viszony alapján csatlakoznak az egyes táblák egymáshoz.

Mellékeltem a lap alján, mert nagyobb fél oldalnál.
I. ábra: EK-diagramm

- *II. ábrán* már azt is pontosan láthatjuk, hogy melyik cellák vannak pontosan összekötve, vagy idegen kulcsként átadva az adatbázisomba.

Mellékeltem a lap alján, mert nagyobb fél oldalnál.
II. ábra: Bachmann ábra

e. Összekapcsolásuk:

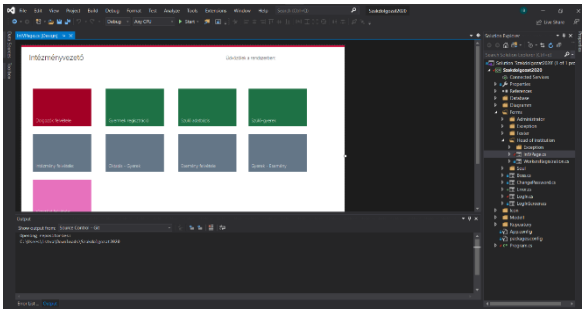
- ALTER TABLE `**children_school_ibfk_1**` ADD CONSTRAINT `children_school_ibfk_1` FOREIGN KEY IF NOT EXISTS (`schoolID`) REFERENCES `school` (`ID`);
- ADD CONSTRAINT `**children_school_ibfk_2**` FOREIGN KEY IF NOT EXISTS (`childrenID`) REFERENCES `children_fullprofile` (`ID`);
- ADD CONSTRAINT `**ceventsk_ibfk_1**` FOREIGN KEY IF NOT EXISTS (`childrenID`) REFERENCES `children_fullprofile` (`ID`);
- ADD CONSTRAINT `**ceventsk_ibfk_2**` FOREIGN KEY IF NOT EXISTS (`eventsID`) REFERENCES `children_events` (`ID`);
- ADD CONSTRAINT `**parentsk_ibfk_1**` FOREIGN KEY IF NOT EXISTS(`childrenID`) REFERENCES `children_fullprofile` (`ID`);
- ADD CONSTRAINT `**parentsk_ibfk_2**` FOREIGN KEY IF NOT EXISTS(`pID`) REFERENCES `parents` (`ID`);
- ADD CONSTRAINT `**children_health_ibfk_1**` FOREIGN KEY IF NOT EXISTS (`childrenID`) REFERENCES `children_fullprofile` (`ID`);

II. Szoftver

Az asztali alkalmazáshoz, ki kell találni elsőként, hogyan szeretném megvalósítani a terveimet. Hiszen anélkül nehezen lehet előre haladni, amíg nincs alap. Fontosnak tartottam, hogy a dolgozok közötti információ áramlás gyorsabb legyen, mint eddig volt. Nagyobb mérföldkőnek gondolom azt, hogy ha egy pszichológus csinál egy felmérést egy gyerekről a legelején, a szoftver segítségével tudja meg ennek az eredményét minél hamarabb, hogy olyan szellembe tudja nevelni. Hogy a gyerek lelki dolgaira is oda tudjon figyelni.

A programnak több rétege is van: a felhasználok azonosítása (kiléphet be), milyen feladatokat lásson el (általában ezt a megrendelővel való egyeztetés során derül ki, és szoros együtt működéssel történik, vagy az interjú segítségével) vagyis a funkcionális[4] és minőségi követelményeket[5] kellett szem előtt tartani. A funkciókat én fektettem le, az interjúk alapján.

Az én választott fejlesztési felületem az a Microsoft Visual Studio 2019 (*III. ábra*). Nagy előnye még a Microoft Visual Studio-nak, hogy több verziókövető rendszert[6] is hozzá lehet kötni, én a GitHub-t alkalmaztam a szakdolgozatomba, de emellett lehet Azure-t is használni. Azért választottam ezt, mert tisztába vagyok a kezelésével és természetesen ingyenes (2019 kiadás). Nagyon sok framware[7] -t lehet hozzá letölteni. Én a Metro framwork-t használom, mint keretrendszer. Fontos megemlíteni, hogy a programozási nyelvem C# lett, mert az iskolai tanulmányaim alatt ezt tanultuk meg olyan szinten, hogy tudjam alkalmazni egy magasabb szinten is. Pozitívnak számít még, ennél a fejlesztési környezetnél, hogy személyre szabható. Itt akár gondolhatunk az egyes ablakok elrendezésére (például Team Explorer, Properties, Solution) és vagy a 'Dark' módra, ahol a szemünknek is kevesebbet árt és több ideig tudjuk használni egy nap leforgássá alatt. 2019-es verzióban már több szín is elősegítik a munka gyorsabb, rendezett előre haladását. Valamint a referenciák elárulják a fejlesztőknek, hol lett használva az adott függvény, metódus. Ez platformhoz kötött, vagyis Microsoft Windows operációs rendszer szükséges hozzá. De van példa platform független programozási nyelvekre, is például Java.



III. ábra: Microsoft Visual Studio 2019 felülette

Továbbá a szoftvernek midig át kell esnie egy tesztelés folyamaton is. Így ki tudjuk az egyes gyenge pontokat szűrni. Az egyes modelleken belül adatok bevitele, módosítása, beolvasása tesztelve van, hogy nem kerüljön olyan adatt az adatbázisba, ami nem belevaló. Egyéb esettben.

[4] Milyen funkciókat várunk el a szoftvertől.

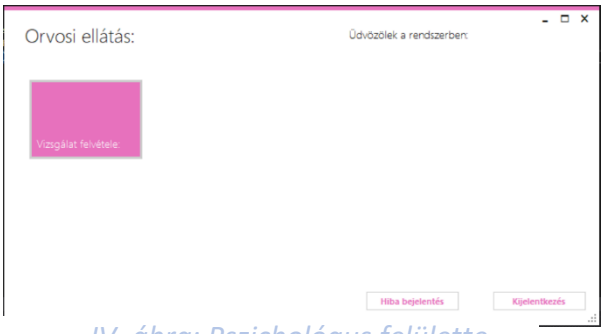
[5] Például ide tartozik: design, válaszidő, könnyen kezelhető legyen így könnyen tanulható is.

[6] A lényege az, hogy nyomon tudod követni a programod alakulását, és akár egy régi állapotba visszaállítani, kommenteket hozzá fűzni és természetesen közös.

[7] Keretrendszer lehet például: Metro frameware (máshogy néz ki a megjelenésben).

Elsődleges cél volt az is, hogy úgy tervezem meg a programot minél felhasználó barátabb, de emellett esztétikus is legyen. Ez azt takarja, hogy intézményben dolgozó emberekkel szoros együtt működés és igények meghallgatása mellett terveztem és alakítottam ki a végső megoldást. Követelményfeltáráshoz elengedhetetlen a leendő felhasználok megkérdezése. Ez által elérve azt, hogy minél egyszerűben meg lehessen tanulni. Így az intézményben dolgozóknak nem kell félni egy újabb program megtanulásától. Így akár könnyebben tudnak egymásnak is segíteni.

Fókuszban volt, hogy legyen letisztult, de mégis szép megjelenése legyen és egyértelművé téve, hogy mit hova kell beírni vagy kiválasztani. Vagy ha az adott elem megnyomásakor mi fog történi.



IV. ábra: Pszichológus felülette

Oda figyeltem arra is, hogy destruktív műveleteket igyekeztem elkerülni, vagyis, ha egy olyan parancs fut le, azt előtte kérdezze meg a felhasználót. Erre tipikus példa a törlés.

A színek használatára nagyobb hangsúly fektettem. Az egyes felhasználok, szerepkörüktől függően (*IV. ábra*) más – más színű a felülete (szürke, piros, rózsaszín, zöld). Az általános ablakok (bejelentkezési ablak) is külön színt kapott.

a. Verziókövetés

A szoftver fejlesztés során fontos a verzió követés, és a sztorik megírása. Ebben a Github segített nekem, hogy nyomon tudja követni, hogy mi az, amivel elkészültem és mi az ami még hátra van, ami még megoldandó feladat. Vagy esetleg még folyamatban van.

b. Felhasználok:

Következők a lehetséges felhasználok hozzáférése a szoftverhez. Minden szerepkör adott részhez fér hozzá.

Használat esett diagram arra jó, hogy lehet látni az egyes joggal bíró felhasználok, mihez tudnak hozzáférni, módosítani, hozzáadni, vagy akár törölni vagy megtekinteni. Live In Care-nél (*III. ábra*) szemlélteti részletesen.

Egy nevelő egy eseményt rakhat fel, ami a gyerekkel történt. Például „Mecseken volt az osztálykirándulás alkalmával”. Ezt feltölti a program segítségével és természetesen gyerek beszámolója alapján (hogy érezte magát, mi az, amit magával visz). Később a szülő ezt a webes felületen meg is tudja nézni.

1) **Intézményvezető** - aki felelős az adott intézményért (igazgató), a dolgozók felvétele, módosítása, törlése. Mindegyik felülethez hozzá fér (teljes jogkör a programban).
2) **Intézmény ügyintéző** - aki azért felelős, hogy a gyerekek teljes profilja[8] , a gyerekek szüleinek adatai felkerüljön az adatbázisba a programon keresztül, és szülő bejelentkezés jogait kezeli a gyereket megkérdezve.

[8] Gyerek minden okmányainak adatai és egyéb információkat tartalmazza.

3) **Pszichológus** - akinek a feladta a gyerekek alaphelyzet felmérése miután bekerült a rendszerbe közvetlen (SNI[9], BTM[10]), valamint az egyéb egészségügyi vizsgálatok rögzítése a rendszerbe.
4) **Nevelőszülő** - aki azzal járul hozzá a rendszer működéshez, hogy egyéb fontos adatok[11] felvételében segíti a munkát.
5) **Szülő** - aki a gyerekkel történt eseményeket[12] és a vizsgálatokat tekintheti meg webes felületen, regisztráció esetén. Ehhez rendelkezniük kell regisztrációs jelszóval és felhasználóval.
6) **Gyerek** - akiről az adatokat feltöltjük (alanya a szoftvernek) és ők segítése az elsődleges célja a programnak.
7) **Admin** – aki azért felelős, ha problémát észlel neki jelezze és a probléma helyzet felmérés utána, a problémát minél hamarabb megoldja. És ha valaki hibát észlel, azt ő kijavítsa/közbenjárjon. Ő segíti a betanítás is a dolgozok számára. Vagy felmerülő kérdések megválaszolására.

Mellékeltem a lap alján, mert nagyobb fél oldalnál.
III. ábra: Use-Case Diagram

c. Gépigény:

A gépigény az, ami leírja azt, hogy mi az, ami feltétlen szükséges egy adott szoftver teljes körű működéséhez. Ami szükséges hardware teljesítmény foglal magában, vagy akkor egy másik szoftver is igényelhet.

1) Microsoft Windows operációs rendszer (Microsoft Visual Studio 2019-hez szükséges Windows 10)
2) Bevitelre alkalmas billentyűzet, egér
3) Egy monitor a megjelenítéshez
4) Hardver követelmény:
    a. 1,8 GHz vagy gyorsabb processzor. Négymagos vagy jobb ajánlott.
    b. 2 GB RAM a minimális; 8 GB RAM ajánlott (legalább 2,5 GB, ha virtuális gépen fut)
    c. Merevlemez-terület: a telepített funkcióktól függően legalább 800 MB – 210GB szabad hely.
    d. Az általános telepítések 20-50 GB szabad helyet igényelnek.
    e. Merevlemez sebessége: a teljesítmény javítása érdekében telepítse a Windows és a Visual Studio egy SSD-re.
    f. Videokártya, amely támogatja a minimális 720p felbontást (1280x720).
    g. A Visual Studio a legjobban WXGA (1366 x 768) felbontással működik.

d. Interjúk:

Egy komplex program tervezése során, elengedhetetlen az interjú elvégzése. Hogy egy nagyobb képet kapjunk az egyes feladatkörökről egy programmal, hogy lehet ezt alátámasztani a jövőben. Emellett ismernünk azokat a fenntartásokat, amiktől az egyes felhasználók félhetnek és erre egy megoldást kell nyújtani a programba. Esetlegesen

[9] Sajátos tanulású gyerek.
[10] Sajátos beilleszkedés igényű gyerek (lelki, egészségügyi eredetű).
[11] Egészségügyi állapot, események, iskolázottság felvétele.
[12] Ez lehet kirándulás, iskolai program vagy más egyéb programleválasztás.

valamilyen elvárást támasztanak egy funkció irányába. „Úgy a leggazdaságosabb, ha ez a cella ott van ez meg emitt.” Vagy hogy a keresés funkció mi alapján szűrjön. Név? Életkor?

Fenntartások:

- **Intézményvezető:** „A szoftver használata és elsajátítása. Több odafigyelés és megfelelés.”
- **Ügyintéző:** „Nincs semmi félelmem, hiszen rengeteg programmal dolgozok napi szinten.”
- **Pszichológus:** „Ne legyen nyilvános az, amit ő felvisz a vizsgálat után. Mert az a nevelőnek szeretném írni, nem másnak.”
- **Gyerek:** „Megfelelő adatokat, információkat raknak fel rólam.”
- **Szülő:** „Nagyon jó ötletnek tűnik, hogy figyelhetem a gyerekem, csak nem vagyok az a nagy netezős/gépes.”
- **Nevelő:** „Egy-két dolgot ha nem értek, azt meg tudom kérdezni?” , „Napra készen tartani az adatokat.”

Válaszok:

- **Intézményvezetőnek:** A Live In Care szoftver úgy lehet elkészítve, hogy szem előtt tartsa azt, hogy a program „beszédes legyen”. Vagyis, ha egy cellát nem töltenek ki, ami kötelező, akkor szól a program, vagy ha valami nem helyesen működik (például nem elérhető az adatbázis), azt is közli. Tejesen felhasználóbarát.
- **Pszichológusnak:** Jelenleg a rendszer úgy van megcsinálva, hogy csak a szülő és a nevelő láthatja ezeket az információkat a gyerekről. Máshoz ez nem fog a program által eljutni.
- **Gyereknek:** Jelenleg a szülő webes bejelentkezéskor, meg tudod nézni, hogy a személyes adataidon kívül mit raktak fel rólad. Természtesen, ha ’Gyerek követés’-t engedélyezed.
- **Szülőnek:** Nem igényel semmilyen nagyobb informatikai tudást. Egy regisztrálást kell elvégezni első bejelentkezés előtt. Ha esetleg ebben nehézséget érezz keresse fel intézmény kollégáit és ők segítenek.
- **Nevelőnek:** A segítséget bármikor tudok/tudunk adni. Akár emailen, akár telefonon. A hiba visszajelzésnél gombnál megtalálod a fontosabb elérhetőségeket.

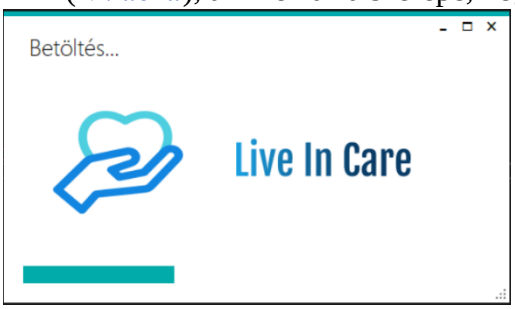
e. Tervezés

Papírforgatókönyvek készítése:

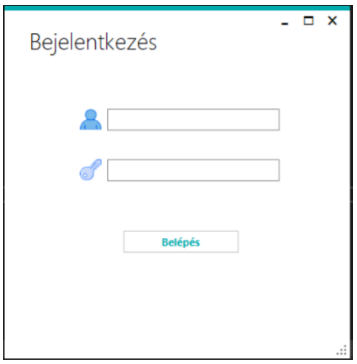
A tervezés egy létfontosságú rész egy program megírása előtt, különösen oda kell figyelni a megjelenésre is. A tervezés során nagyobb kereteket terveztem meg, nem egy pontos előre kigondolt tervet. A megvalósítás során kellett változtatni, eltérni az eredeti tervtől. Ez nem negatív része a tervezésnek, csak Metro Framework-kal sok új kapu nyílt meg, amire tervezéskor nem gondoltam. De a kereteket szépen megadta, és segítségével tudtam, hogy mit szeretnék majd a megvalósítás résznél csinálni. Így volt egy erős terveztem, ami alapján dolgoztam. Amit webes tervező programmal terveztem meg. Hogyha majd erre meg arra kattintok minek kellene majd történie.

f. Bejelentkezés:

A felhasználó bejelentkezés előtt egy töltő képernyőt lát (*IV. ábra*), aminek az a szerepe, hogy a program ekkor hozza létre az adatbázist. Azért ilyenkor megy végbe, hogy a bejelentkezést ez ne lassítsa egy gyengébb gépnél és természetesen azért is, mert ebben az adatbázisban van egy tábla, ami a bejelentkezést vizsgálja, hogy ki léphet be és ki nem (employes_login) a proramba. Betöltés alatt táblákat létrehozza, beszúrja a teszt adatokat, idegen kulcsokat ad hozzá és egyedi jelzőket is.



IV. ábra: Töltő képernyő

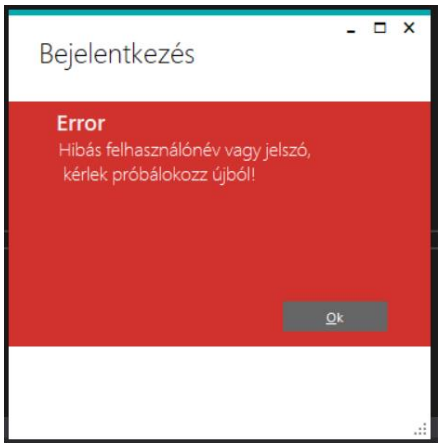


V. ábra: Bejelentkezési felület

Ezután jelenik meg a bejelentkezési lehetőség. Ahol látható két input (textbox) mező.

```
if (dr["epassword"].ToString() == "abc123")
{
    fnameLoged = dr["ename"].ToString();
    userId = dr["ID"].ToString();
    etype = dr["ejob"].ToString();
    ChangePassword cp = new ChangePassword();
    cp.Show();
}
else
{
    etype = dr["ejob"].ToString();
    switch (dr["ejob"].ToString())
    {
        case "Rendszergazda":
            fnameLoged = dr["ename"].ToString();
            Boss f = new Boss();
            f.Show();
            break;
        case "Ügyintéző":
            fnameLoged = dr["ename"].ToString();
            HomePageUgy f2 = new HomePageUgy();
            f2.Show();
            break;
        case "Intézményvezető":
            fnameLoged = dr["ename"].ToString();
            IntVPage ivp = new IntVPage();
            ivp.Show();
            break;
        case "Nevelő":
            fnameLoged = dr["ename"].ToString();
            Fosterhomepage sc = new Fosterhomepage();
            sc.Show();
            break;
        case "Pszichológus":
            fnameLoged = dr["ename"].ToString();
            SoulForm sf = new SoulForm();
            sf.Show();
            break;
        default:
            hiba += "Nem rendelkezik státusszal!";
            labelError.Text += hiba;
            break;
    }
}
```

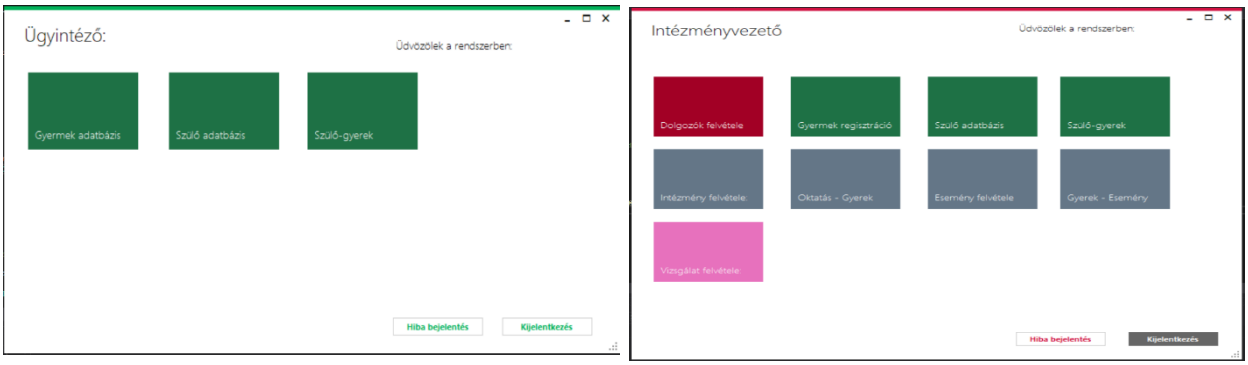
A bejelentkezés ez „if” teszi lehetővé. Ha hibás a jelszó vagy a felhasználónév akkor egy hiba üzenetet ír ki: „Hibás felhasználónév vagy jelszó, kérlek próbálkozz újból!”. Ekkor a beírt jelszó vagy felhasználónév hibás, tehát az adatbázisban nem szerepel ez a páros.



VI. ábra: Hibás bejelentkezés és if ág a bejelentkezéshez

A *VI. ábrán* az is látható, hogy ha a jelszónak alapértelmezett jelszó van beállítva (abc123), akkor kér egy jelszó változtatást is. A felhasználónevet automatikusan a névből generálja. Egy például egy felhasználó névre: Zuhany Rózsa -> zuhroz. A felhasználó és a jelszó is ’Case sensitivity’, vagyis számít a nagy és a kis betű.

A *VI. ábrán* az is szerepel, hogy ha sikeres a bejelentkezés, akkor attól függően, hogy milyen beosztásba vagy a munkahelyen a szerinti felületet nyitja meg. Ezt bejelentkezéskor közvetlen ellenőrzi. Adatbázisban 'employes_login.ejob' oszlopban található. Az asztali alkalmazásban ezt az oszlopot nézi meg a bejelentkezett felhasználó név sorával megegyező cellát. Bejelentkezéshez erre az egy táblára van szükség.



VII. ábra: Ügyintéző (zöld) és Intézményvezető (piros) felülete

g. Felépítése, mappa szerkezet

A mappa struktúrát úgy terveztem meg, hogy a programozás közben minél kézen fekvőbb legyen. Ez hatással volt arra, hogy átlássam és gyorsabban tudjak programozni.

- Form mappa
  - Ebben található az összes form.
  - A formok úgy vannak rendezve, ahogy hozzáférnek jogokban az egyes felhasználók. Például a Soul (pszichológus) mappában van az egészségügyi vizsgálatok regisztráció formja és a bejelentkezés utáni egészségügyi vizsgálat homepage.
- Modell mappa
  - Ezekben található az összes regisztrációs egyedek modellje. Itt vannak az ellenőrzések, getterek, setterek. Itt van beszúráshoz, módosításhoz, kiolvasáshoz MySQL statement[13] parancssorok.
- Repository mappa
  - Itt található az adott egyedhez tartozó összes listás eljárások, és DataGridView formázások egy része.
- Exception mappa
  - Adott classok Exception-jei.

h. Hozzáadás, módosítás, törlés:

- Az asztali alkalmazáson 9 féle regisztrációt lehet elvégezni. Ezek a következők:
  - Ügyintéző
    - Gyerek teljes profiljának beregisztrálása (személyes adatok)
    - A gyermek szüleinek a regisztrációja
    - Valamint a gyerek szülő összekapcsolása, hogy a rendszer lássa hogy ők rokonok.
  - Nevelő

[13] MySQL statement - egy adatbázisos lekérés

        - Oktatási intézmény regisztrációja
        - Adott oktatási intézménnyel jogviszonyba került fiatal összekapcsolása
        - Esemény felvétele
        - Adott eseményen részt vett fiatal felvétele (szülő megnézheti)
    - Intézmény vezető:
        - A dolgozók felvétele az adatbázisba
    - Pszichológus
        - Az egészségügyi vizsgálatok felvétele, amin részt vett a fiatal (szülő megnézheti)

A gyermek regisztrációja:

A gyermek regisztrálása által szeretném bemutatni a felvételt, módosítást és törlést. Minden esettben így használom, maximum egy-két részben tér el. Eltérés az csak, hogy egyes regisztrációknál más-más adatokat kérek be.

Ha a bejelentkezés sikeres volt és természetesen a mi jog körünkbe tartozik (ügyintéző, intézményvezető) a gyermek regisztrációja, akkor már regisztrálható is fel a gyerek. Ezt metroTile grafikai elem fogja jelezni, hogy van -e jogosultsága az adott felhasználónak.

VIII. ábra: metroTile

A regisztrációt objektum orientáltan végzem el, két fő része van. Ez minden regisztráció esetén elmondható. Elsőként az adatbázisban, majd a listában adom hozzá (lehet szülő, gyerek). Ezt azért ebben a sorendeben végeztettem el mert ha van 'egyedi' kulcs, akkor a listához ne adja hozzá, ha az adatbázis nem engedi, mert például két ugyan olyan személy igazolvány és TAJ szám párt nem lehet felregisztrálni. Hiszen akkor adat ismétlés lenne esetlegesen.

- **A gyermek hozzáadása**

A getnextChildId() arra kell, hogy a listát megnézze mi a következő index (ID). A lista adatait megszámolja (.Count()) és egyet hozzáad, ha első akkor eggyel kezdi el indexet.

```
try
{
    int id = repo.getnextChildId();

    Child newEmployee = new Child(
        id,
        metroTextBoxName.Text,
        metroComboBoxSex.Text,
        metroTextBoxIdCard.Text,
        metroTextBoxTaj.Text,
        metroDateTimeBDate.Text,
        metroTextBoxBPlace.Text,
        metroDateTimeComing.Text,
        metroTextBoxLocation.Text
       );

    //Hozzáadás az adatbázishoz
    try
    {
        rep.insertChildrenToDatabase(newEmployee);
    }
    catch (Exception)
    {

        throw new insertChildException();
    }

    //Hozzáadás a listához
    try
    {
        repo.addChildToList(newEmployee);
    }
    catch (Exception)
    {

        Debug.WriteLine("A gyerek felvétele sikertelen volt a listához");
        MetroMessageBox.Show(this, "\n\nHibát észleltünk, a felvétel sikertelen volt a listához.", "Felhívás", MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }

    //DataGridView frissítése
    updateDataInDataGriedViewt();
    updateCildrenNumber();
    emptyCells();
}
```

IX. ábra: Gyermek hozzáadása

- **Adatbázishoz hozzáadása**

A gyermek hozzáadása csak akkor történik meg, ha bevitt adatok helyesek. Ezt minden regisztrációnál a modell-ben ellenőrzőm, hogy minden helyesen lett -e beírva az input mezőkben. Például TAJ szám az 9 db szám lehet csak. Ettől nem lehet eltérni. Minden regisztráció esetben az adatbázishoz a következő szerint adom hozzá. Ebben az esetben Child adatait, ha minden adat helyesen lettek megadva.

```
public void insertChildrenToDatabase(Child newChild)
{
    MySqlConnection connection = new MySqlConnection(connectionString);
    try
    {
        connection.Open();
        string query = newChild.getInsert();
        MySqlCommand cmd = new MySqlCommand(query, connection);
        cmd.ExecuteNonQuery();
        connection.Close();
    }
    catch (Exception e)
    {
        connection.Close();
        Debug.WriteLine(e.Message);
        Debug.WriteLine("InsertChild**********" + newChild + " gyermek beszúrása adatbázisba nem sikerült.");
        throw new RepositoryChildException("Sikertelen beszúrás az adatbázisból.");
    }
}
```

IX..I ábra: Gyermek hozzáadása adatbázishoz (insertChildrenToDatabase)

A C#-be beépített MySQL kódokat használom. Ezt bővítményként hozzá kell adni, hogy lehessen használni. A getInsert() az egy MySQL INSERT INTO statement van benne (*X. ábra*).

```
public string getInsert()
{
    return
        "INSERT INTO `children_fullprofile` (`ID`, `cname`, `csex`, `cidcardnumber`, `ctajnumber`, `cbirth`, `cbirthplace`, `ccoming`, `clocation`)"
        +"VALUES ('"
        +cID
        +"', '"
        +getCcname()
        +"', '"
        +getCsex()
        +"', '"
        +getCidcard()
        +"', '"
        +getCtajNumber()
        +"', '"
        +getCbirthday()
        +"', '"
        +getCbirthdayplace()
        +"', '"
        +getCcoming()
        +"', '"
        +getClocation()
        +"');";
}
```

X. ábra: Gyermek hozzáadása (getInsert)

A gyerek (Child) modell osztállyal partial-ba van, vagyis megosztottan. Vagyis mintha egy fájl lenének, de mégis fizikálisan külön fájlban vannak. Ezt az asztali programozásom során sokszor alkalmaztam, mert így egy egységbe tartozó metódusokat, függvényeket tudtam egy fájlba rakni, de ettől egy osztályba vannak. Így könnyebben átlátható és könnyebben megtalálható egyes részei a programnak.

- **Listához hozzáadása**

```
List<Child> children;
```

```
public void setChild(List<Child> children)
{
    this.children = children;
}
```

XI. ábra: Gyermek lista és a setter-e

XI. ábrán a gyermekeknek a listája látható, amibe tárolom a gyerekek adatait. Például egy gyerek nevét, születési idejét, TAJ számát és még sok mást. Setter (setChild()) arra kell, hogy a tesz adatokat betöltse a listába, amit külön lekérünk és ennek adjuk át. Természetesen arra is, hogy a regisztrációs form-on az adatok elérhetőek legyenek.

```
public void addChildToList(Child newChild)
{
    try
    {
        children.Add(newChild);
    }
    catch (Exception e)
    {
        throw new RepositoryChildExceptionCantAdd("Nem lehet új gyermeket hozzáadni a listához!");
    }
}
```

XII. ábra: Gyermek hozzáadása a listához (addChildToList)

XII. ábra azt mutatja, hogy adom hozzá a listához Child, ha helyesen lett minden információ beírva a regisztrációs form-on.

- **A gyermek módosítás**

```
try
{
    Child modified = new Child(
    Convert.ToInt32(metroTextBoxID.Text),
    metroTextBoxName.Text,
    metroComboBoxSex.Text,
    metroTextBoxIdCard.Text,
    metroTextBoxTaj.Text,
    metroDateTimeBDate.Text,
    metroTextBoxBPlace.Text,
    metroDateTimeComing.Text,
    metroTextBoxLocation.Text
    );
    int id = Convert.ToInt32(metroTextBoxID.Text);

    //Módosítás az adatbázisban
    try
    {
        rep.updateChildrenInDatabase(id, modified);
    }
    catch (Exception ex)
    {
        throw new updateChildException();
    }

    //Módosítás a listában
    try
    {
        repo.updateChildInList(id, modified);
    }
    catch (Exception ex)
    {

        Debug.WriteLine("A gyermek módosítása sikertelen volt
        MetroMessageBox.Show(this, "\n\nHibát észleltünk, a m
    }


    //Módosítás miatt DataGridView updatelése
    updateDataInDataGriedViewt();
}
```

XIII. ábra: Gyermek módosítása

A gyermek módosítása ugyan úgy működik, mint szinte a hozzáadás. Csak helyes adatok megadásával lehetséges és ha egy sort kiválasztott, amit módosítani szeretne. Először az adatbázisba történik hiszen megint nem lehet két ugyan olyan TAJ szám és Személyigazolványszám pár. Az indexet itt nem kell ellenőrizni, hiszen egy részt a felhasználó nem tudja azt módosítani, más részt a módosítandó sor indexe ugyan az marad. Csak át kell alakítani egész számmá. Módosítás egy adott sor kiválasztásával lehetséges. Később a felhasználói dokumentáció részben részletesen leírom.

- **Adatbázisban módosítás**

```
public void updateChildrenInDatabase(int id, Child modified)
{
    MySqlConnection connection = new MySqlConnection(connectionString);
    try
    {
        connection.Open();
        string query = modified.getUpdate(id);
        MySqlCommand cmd = new MySqlCommand(query, connection);
        cmd.ExecuteNonQuery();
        connection.Close();
    }
    catch (Exception e)
    {
        connection.Close();
        Debug.WriteLine(e.Message);
        Debug.WriteLine("UpdateChild**************************" + id + " idéjű dolgozó módosítása nem sikerült.");
        throw new RepositoryChildException("Sikertelen módosítás az adatbázisból.");
    }
}
```

XIV. ábra: Gyermek módosítása adatbázisban (updateChildrenInDatabase)

Hozzáadáshoz hasonlóan itt is, egy MySQL-s UPDATE statement-et használok, getUpdate() (*XV. ábra*). Itt az id alapján update-el, mert azt nem lehet megváltoztatni. Legalább is a felhasználó nem tudja, így ez a kézen fekvő.

```
public string getUpdate(int id)
{
    return
        "UPDATE `children_fullprofile` SET `cname`= '"
        + getCcname()
        + "', `csex` = '"
        + getCsex()
        + "', `cidcardnumber` = '"
        + getCidcard()
        + "', `ctajnumber` = '"
        + getCtajNumber()
        + "', `cbirth` = '"
        + getCbirthday()
        + "', `cbirthplace` = '"
        + getCbirthdayplace()
        + "', `ccoming` = '"
        + getCcoming()
        + "', `clocation` = '"
        + getClocation()
        + "' WHERE `children_fullprofile`.`ID` = "
        + id + ";";
}
```

XV. ábra: Gyermek módosítása adatbázisban (getUpdate)

- **Listában módosítás**

```
public void updateChildInList(int id, Child modified)
{
    Child chi = children.Find(x => x.getCID() == id);
    if (chi != null)
    {
        chi.updateL(modified);
    }
    else
    {
        throw new RepositoryChildExceptionCantMoodify("Nem lehet modósitani a listában a gyermeket!");
    }
}
```

XVI. ábra: Gyermek módosítása listában (getUpdate)

A updateChildInList() függvényben egy id alapján módosít. Ezt úgy teszi, hogy megkeresi a listában az adott id-jű gyereket getCID() segítségével. Majd updateL() segítségével adott gyerek adatait frissíti az új adatokkal.

```
public void updateL(Child modified)
{
    this.cname = modified.getCcname();
    this.csex = modified.getCsex();
    this.cidcard = modified.getCidcard();
    this.ctajnumber = modified.getCtajNumber();
    this.cbirthday = modified.getCbirthday();
    this.cbirthdayplace = modified.getCbirthdayplace();
    this.ccomming = modified.getCcoming();
    this.clocation = modified.getClocation();
}
```

XVII. ábra: updateL(), gyerek adatok módosítása

- **A gyermek törlése:**

```
try
{
    int selectedIndex = metroGridChildren.SelectedRows[0].Index;
    DialogResult dr = MetroMessageBox.Show(this, "\n\nBiztos szeretné törölni a gyermeket?", "Dolgozó törlése", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
    if (dr == DialogResult.Yes)
    {
        int id = -1;
        if (!int.TryParse(metroGridChildren.SelectedRows[0].Cells[0].Value.ToString(), out id))
        {
            return;
        }

        //Törlés az adatbázisból
        //Törlés a listából
        try
        {
            rep.deleteChildFromDatabase(id);
            repo.deleteChildInList(id);
        }
        catch (Exception ex)
        {
            Debug.WriteLine("Sikertelen törlés! Nem lehet törlni a gyereket, mert másik táblában is szerepel.");
            MetroMessageBox.Show(this, "\n\nSikertelen törlés! Nem lehet törlni, mert másik táblában is szerepel.", "Felhívás", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }

        //DataGridView frissítése
        updateDataInDataGriedViewt();
        updateCildrenNumber();
    }
    else
    {
        Debug.WriteLine("'DialogResult.No'-ra futott rá!");
    }
}
```

XVIII. ábra: Gyermek törlése

A gyermek törlése is úgy működik, mint a módosításnál. Ki kell egy sort választani majd a törlés gombra kattintani.

- **Adatbázisban törlése**

```
public void deleteChildFromDatabase(int id)
{
    MySqlConnection connection = new MySqlConnection(connectionString);
    try
    {
        connection.Open();
        string query = "DELETE FROM children_fullprofile WHERE ID=" + id;
        MySqlCommand cmd = new MySqlCommand(query, connection);
        cmd.ExecuteNonQuery();
        connection.Close();
    }
    catch (Exception e)
    {
        connection.Close();
        Debug.WriteLine(e.Message);
        Debug.WriteLine("DeleteChild**********" + id + " idéjű gyermek törlése nem sikerült.");
        throw new RepositoryChildException("Sikertelen törlés az adatbázisból.");
    }
}
```

XVIII. ábra: Gyermek törlése adatbázisból

A törlés id alapján működik. Az adott táblában indexét megfeleltetjük azzal index-szel, amit kiválasztottunk sor és azt adjuk át törléskor. Így törli az adatbázisból, ehhez MySQL DELETE statement-t használok.

- **Listában törlés**

```
public void deleteChildInList(int id)
{
    Child chi = children.Find(x => x.getCID() == id);
    if (chi != null)
    {
        children.Remove(chi);
    }
    else
    {
        throw new RepositoryChildExceptionCantDelete("Nem lehet törölni a gyermeket a listából!");
    }
}
```

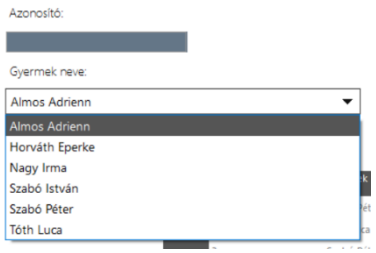
XIX. ábra: Gyermek törlése listából

Az adott id-jű gyereket megkeresi a listában, majd a beépített C#-os Remove()-val törli a listából.

Eltérés lehet a módosítás és a hozzáadás az egyes formokon belül. Azért, mert van olyan form, amihez például a gyerekek neveit kellett ki gyűjtenem a listából és egy comboboxba feltöltenem a neveket, ami a gyerekek regisztrációs fromjához tartozott. De adatbázisba a gyerek indexét kellett beszúrnom, még a listában a nevét illesztettem be, hogy a megjelenéskor a neve látszódjon és ne az indexe.

```
EventChild newEventChild1 = new EventChild(
    id,
    metroComboBoxChildrenName.Text,
    metrocomboboxEvent.Text,
    metroDateTimeeventDate.Text
   );

EventChild newEventChild2 = new EventChild(
    id,
    rc.getChildrenId(metroComboBoxChildrenName.Text),
    re.getEventID(metrocomboboxEvent.Text),
    metroDateTimeeventDate.Text
   );
```

```
//Hozzáadás az adatbázishoz
try
{
    rce.insertEventChildrenToDatabase(newEventChild2);
}
catch (Exception)
{
    throw new insertEventChildrenException();
}

//Hozzáadás a listához
try
{
    rec.addEventChildrenViewToList(newEventChild1);
}
```

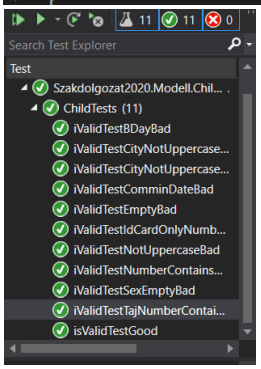


XXIV. ábra: Adatbázis/lista hozzáadás, módosítás

XXIV. ábrán az látható a legürdülő menüből kiválasztom a nevet, de nem azt szúrom be egy hozzáadás alkalmával, hanem a kiválasztott név indexét, erre jó getChildrenId() eljárás.

i. Unit teszt

```
public bool isValid()
{
    if (!isValidName(cname))
    {
        throw new ModellChildNotValidNameException("Nem megfelelő 'Neve' mező, kezdje nagy betűvel a nevet és legalább 4 karakter hosszú
          legyen. Számot ne tartalamzzon. Kérlek próbáld újra!");
    }
    if (!isValidSex(csex))
```




```
[TestMethod()]
0 references
public void iValidTestTajNumberContainsLetterBad()
{
    try
    {
        Child ss = new Child(1, "Szabó Péter", "férfi", "785962LA", "79963AA65", "2005-12-04", "Szeged", "2008-06-01", "Szeged");
        if (ss.isValid())
        {
            Assert.Fail("Betűt tartalmaz a taj szám esetén nem dob kivételt!");
        }
    }
    catch (ModellChildNotValidTajnumberException mce)
    {
        if (mce.Message != "Nem megfelelő 'Taj szam:' mező. Kérlek figyelj oda a formátumra, csak számot tartalmazhat. pl:123456789")
            Assert.Fail("Betűt tartalmaz a taj szám esetén hibát dob, de a hiba szövege rossz!");
    }
    catch (Exception e)
    {
        Assert.Fail("Betűt tartalmaz a taj szám esetén nem dob kivételt vagy rossz kivételt dob!");
    }
}
```

XXIV.I . ábra: Unit teszt

A szoftver készítéshez elengedhetetlen az egység teszt, ezzel is megnézve a bemeneti adatok helyesek -e, vagyis a specifikációk. Az egység tesztet (unit teszt), a gyerekek adatain végeztem el, mert gyerekek adatai több szempontból is fontosak, egyrészt más formok is használják a gyerekek adatait. Másrészt a weboldal is egy részét használja. A gyerek adatait minden adatát teszteltem. Ami a Hibakezelésben van *XX. ábra* azokat a validációkat teszteltem le. Minden megfelelően működött. Egyik példát említve azt teszteltem le, hogy a taj számba írtam betűket (*XXIV.I. ábra*). Ami csak 9 db számot tartalmazhat. De tesztelve lett a gyerek neve, születési ideje, születési helye, tartózkodási helye, személyigazolvány száma, belépés ideje.

j. Hibakezelés:

Igen fontos a hibakezelés és a felhasználó értesítése, hogy éppen miért nem jó, amit csinál vagy mi a kiváltó hiba oka. Egyik legfontosabb része a hiba kezelésemnek az a különböző regisztrációs felületek bevitt adatok ellenőrzése. Itt is a gyerek regisztráció által fogom ezt bemutatni, mert mindegyiknél így van. Modellbe nézem meg az adott adatok bevitele helytáló -e (például személyigazolvány szám, 6 szám elől majd két nagy betű).

```
public Child(int cID, string cname, string csex, string cidcard, string ctajnumber, string cbirthday, string cbirthdayplace, string ccomming, string clocation)
{
    if (!isValidName(cname))
    {
        throw new ModellChildNotValidNameException("Nem megfelelő 'Neve' mező, kezdje nagy betűvel a nevet és legalább 4 karakter hosszú legyen. Számot ne tartalamzzon. Kérlek próbáld újra!");
    }
    if (!isValidSex(csex))
    {
        throw new ModellChildNotValidSexException("Nem megfelelő 'Neme' mező, kattintson a lefele mutató nyilra 'Neme' mezőnél és a legördülő menő segítségével válasza ki a nemet.");
    }
    if (!isValidIdCard(cidcard))
    {
        throw new ModellNotValidChiIdcardException("Nem megfelelő 'Személyigazolvány szám:' mező. Kérlek figyelj a formátumra. Pl: 123456AB");
    }
    if (!isValidTaj(ctajnumber))
    {
        throw new ModellChildNotValidTajnumberException("Nem megfelelő 'Taj szam:' mező. Kérlek figyelj oda a formátumra, csak számot tartalmazhat. pl:123456789");
    }
    if (!isValidBirthDay(cbirthday))
    {
        throw new ModellChildNotValidBirthdayException("Nem megfelelő 'Születési idő:' mező, kérem állítsa be a felvett személy születési dátumát!");
    }
    if (!isValidBirthPlace(cbirthdayplace))
    {
        throw new ModellChildNotValidBirthplaceException("Nem megfelelő 'Születési hely' mező, kezdje nagy betűvel a város nevet. Kérem próbálja újra.");
    }
    if (!isValidComing(ccomming))
    {
        throw new ModellChildNotValidComingException("Nem megfelelő 'Belépés időpontja:' mező, kérem állítsa be a felvett személy belépési dátumát!");
    }
    if (!isValidLocation(clocation))
    {
        throw new ModellChildNotValidLocationException("Nem megfelelő 'Intézmény helye, ahol tartózkodik:' mező. Kezdje nagy betűvel a város nevét, irányító számot ne írjon. Kérem próbálja újra.");
    }
```

```
    //***********Set************
    this.cID = cID;
    this.cname = cname;
    this.csex = csex;
    this.cidcard = cidcard;
    this.ctajnumber = ctajnumber;
    this.cbirthday = cbirthday;
    this.cbirthdayplace = cbirthdayplace;
    this.ccomming = ccomming;
    this.clocation = clocation;
}
```

XX. ábra: Gyerek adatainak ellenőrzése a modellben

Ahogy a *XX. ábrán* látható, először leellenőrzöm, hogy gyerekről az információk helyesen lettek -e megadva, így a teszt adatok is, le lesznek ellenőrizve miközben a listához hozzáadja a program indulásakor. Ekkor dobnak Exception-t, ha nem felelnek meg a feltételeknek.

```
public bool isValidLocation(string name)
{
    if (name == string.Empty)
    {
        return false;
    }
    if (!char.IsUpper(name.ElementAt(0)))
    {
        return false;
    }
    if (name.Length <= 5)
    {
        return false;
    }
    return true;
}
```

XXI. ábra: Gyerek adatainak egyik feltétele

XXI. ábra az egy példa egy ellenőrzésre, amit a gyerek konstruktorában hívunk meg. Ha ez true-ként tér vissza akkor a bevitt adatt helyes, ha false-ként tér vissza, akkor dobja a hibát. Hiszen nem helyes a bevitt adatt.

XXII. ábra: Hiba jelzése rossz bevitt adatnál

A *XXII. ábra* mutatja, hogy ha rákattintasz a felvétel vagy a módosítás gombra, de üres vagy nem megfelelő a bevitt adatt akkor egy piros alapon egy fehér kereszt jelenik meg villogva. Ha ráviszed a kurzorod akkor, ki írja a pontos hibát. Ez a modell-ből jön, lásd *XX. ábra*.

Törlésnél szintén van hibakezelés, ha nem jelölt ki egy sort se és úgy akar törölni, vagy ha olyan sort akar törölni, ami más táblában is van.

Ha adatbázis nem elérhető (*XXIII. ábra*) akkor is figyelmezteti az adott felhasználót a metroTile-ra (*VIII. ábra*) kattintás során.

```
private void metroTileAddMEvents_Click(object sender, EventArgs e)
{

    try
    {
        SoulReg sr = new SoulReg();
        sr.Show();
    }
    catch (Exception ex)
    {

        Debug.WriteLine(ex.Message);
        MetroMessageBox.Show(this, "\n\nHibát észleltünk! Az adatbázis nem érhető el, vagy a bemeneti adatt nem megfelelő. Kérem próbálja újra később!", "Hiba", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
}
```

XXIII. ábra: Nem elérhető adatbázis

k. Összefüggés a weblappal

Az asztali alkalmazáshoz tartozik az is, hogy a szülőket beregisztrálja. A szülők beregisztrálása a következő képen zajlik.

XXIV. ábra: Szülő regisztráció

A szülő regisztrálásához *XXIV. ábra* mutatja mi szükségesek. Csak a fehér mezőket tudja az ügyintéző kitölteni. A többi automatikusan generálódik. Ez a többi formnál is így van. Miután az adatokat helyesen beírtuk. A program generál egy regisztrációs jelszót és egy felhasználót, amit az eredeti név megadásból generálja. A regisztrációs jelszó minden esetben 'abc123' lesz. Ezzel a regisztrációs párral tud majd a szülő bejelentkezni. De van egy belépési engedélyezés („Engedélyezi a szülő bejelentkezését:" cella név), amit a gyerek dönt el, hogy szeretné -e, hogy a szülei lássa. Helyzettől függ, hogy engedélyezve van vagy sem. Mert ha a gyereket bántalmazták otthon, akkor nyilván tiltva lesz. Bármikor lehet ezen módosítani.

l. Fejlesztési lehetőség

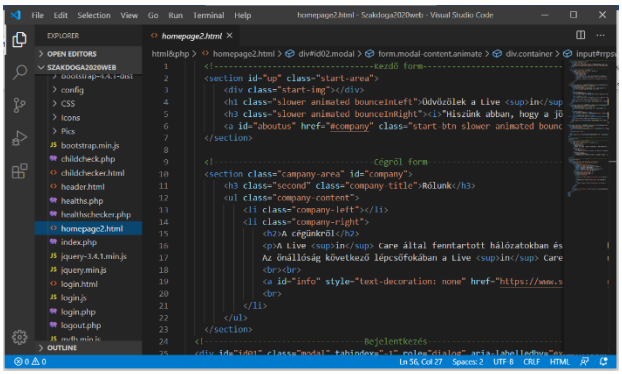
Vannak olyan tervek, amiket sajnos az idő hiányába nem tudtam megvalósítani. De szükségét érzem, hogy benne legyen. Ez azért lenne fontos, hogy későbbiekben egy cél közönség ezt ki tudja próbálni, akár egy béta verzióba. De ehhez egy-két funkciót ki kellene fejleszteni, amit a közel jövőben meg is fogok tenni.

- Vizsgálat hozzáadásakor, legyen szabályozható, hogy mit lát a szülő belőle mit nem a weben. Az egyes párok hozzáadása (például gyerek-esemény, gyerek-iskola) egy formon legyen ne külön. A bekerülés okai, körülményeinek felvétele egyes gyerekeknél. Jogok szabályzása, egy felhasználó több dologhoz is hozzáférhessen, ha úgy szeretnék beállítani. A szül regisztrációnál biztonságosabb jelszó generálás. Hiba jelentés elküldése a programmal

III. A webalkalmazás

a. Bevezetés, probléma felvetés

A webfejlesztés során Visual Studio Code -t használtam (*XV. ábra*), mert tanulmányaim alatt ez a program segítette elő azt, hogy gyorsan tudjak dolgozni és hatékonyan. Fontos megemlíteni, hogy ennek van, 'Dark' módja is, ami kevésbé rongálja szemem több órás munka mellett. Külön pozitív a programban, hogy mappa szerkezetet is meg lehet vele nyitni, így egyszere több fájlal is tudok dolgozni egy időben.



XXV.ábra: Visual Studio Code

Visual Studio Code -ban .js, .php, .css, .htlm fájlkiterjesztésű fájlokkal dolgoztam. Külön fontos volt számomra a választáskor, hogy külön színeket használ az egyes kiterjesztésű fájloknál. Kiterjesztést is könnyen lehetet változtatni. Nem kellett egy felugró ablakot erre használni, csak kiterjesztést átírni. Könnyen kezelhető és sok segítséget tartalmazz, akár a kiegészítésre gondolunk, vagy a hiba keresésre.

Az interjúk során, az derült ki számomra, hogy azok a gyerekek, akik bekerülnek egy állami gondoskodásba általában nem sokat tudnak a szülőkkel lenni, mert ők elmennek dolgozni, hogy minél hamarabb ki tudják a gyereküket hozni. Vagy a távolság olyan nagy a lakhelye / hajléktalan szálló között, hogy nem tudnak túl gyakran találkozni a gyerekükkel.

Valamint arra jöttem rá, hogy a szülők annak is örülnének, ha tudnák, hogy a gyerekük milyen állapotba van mind lelkileg mind fizikálisan. Mert állami gondoskodásba élő gyerekeknek van pszichológia elbeszélgetésük fél évente. Abból a szülő is tudná, hogy minden rendben van -e a gyereke. Első az egészség. De ebbe benne lenne más vizsgálatok is például szemvizsgálat, vér teszt és más is.

A szakdolgozatom ezt a részét fedné le. Hogy a szülőknek, ha nincs idejük a gyerekükkel lenni a munkahely miatt, vagy távolság miatt ez segíthet. Nyilván ez nem tudja a látogatást helyettesíteni, csak jobb oda figyelést terem a szülők irányából.

b. Felhasználók

Mellékeltem a lap alján, mert nagyobb fél oldalnál.

I.ábra: Use-Case diagramm

I. ábra mutatja az összes felhasználót és hogy adott felhasználok milyen jogokkal rendelkeznek. A webes a szülő részre korlátozódik. Hatással van rá más felhasználó is.

c. Csatlakozás adatbázishoz

Csatlakozást az adatbázishoz a config.php végzi. A menü sáv változtatást és be van -e jelentkezve a function.php ellenőrzi. Ezeket hívom meg a require_once-sal php fájlok előtt, csak init.php -ba van mind kettő berakva (*require_once('config/init.php');*).

d. Homepage (kezdő lap):

A kezdő lapot 3 részbe szedtem szét. A head, menu részét külön html fájlba raktam és body részt megint külön. Azért így csináltam meg, mert egyes adatok lekérdezésénél a head-t az adott php fájlba meghívtam, hogy elérje a hivatkozásokat például jquery-3.4.1.min.js vagy például style.css -t formázás szempontból.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="bootstrap-4.4.1-dist/css/bootstrap.min.css">
    <script type="text/javascript" src="jquery-3.4.1.min.js" ></script>
    <script type="text/javascript" src="bootstrap.min.js" ></script>
    <script type="text/javascript" src="mdb.min.js" ></script>
    <script type="text/javascript" src="login.js" ></script>
    <script type="text/javascript" src="popper.min.js" ></script>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" href="CSS/style.css">
    <link rel="stylesheet" href="CSS/animate.css">
    <title>Live in Care ❤</title>
    <link rel="shortcut icon" type="image/x-icon" href="icons/icon.png" />

</head>
<body>
```

XXVI. ábra: header.html

A menü 3 részből áll. Egyik az, amikor bejelentkeztél már (*XXVIII. ábra*), mikor még nem jelentkeztél be (*XXVII. ábra*) és valamint, amikor Gyerek követésre, vagy Ellátás menüre kattintottál (*XXIX. ábra*). Menüt style.css-sel formáztam meg.

```
<header>
    <h2><a href="index.php" style="text-decoration: none">Live <sup>in</sup> Care</a></h2>
    <nav>
        <li><a  href="index.php" style="text-decoration: none">Kezdőlap</a></li>
        <li><a  id="color" onclick="document.getElementById('id02').style.display='block'"> Regisztráció</a></li>
        <li><a  href="#quote" style="text-decoration: none"> Idézetek</a></li>
        <li><a  href="#company" style="text-decoration: none"> Rólunk</a></li>
        <li><a href="#const" style="text-decoration: none"> Kapcsolat</a></li>
        <li><a  id="color" onclick="document.getElementById('id01').style.display='block'"> Bejelentkezés</a></li>
    </nav>
</header>
```

XXVII. ábra: menu.html

```
<header>
    <h2><a href="index.php" style="text-decoration: none">Live <sup>in</sup> Care</a></h2>
    <nav>
        <li><a href="index.php" style="text-decoration: none">Kezdőlap</a></li>
        <li><a href="#quote" style="text-decoration: none"> Idézetek</a></li>
        <li><a href="#company" style="text-decoration: none"> Rólunk</a></li>
        <li><a href="#const"  style="text-decoration: none"> Kapcsolat</a></li>
        <li><a id="childrenbtn" href="childcheck.php" style="text-decoration: none"> Gyerek követés</a></li>
        <li><a id="childrenbtn" href="healthschecker.php" style="text-decoration: none"> Ellátás</a></li>
        <li><a href="logout.php" style="text-decoration: none"> Kilépés</a></li>
    </nav>
</header>
```

XXVIII. ábra: menuin.html

```
<header>
    <h2><a href="index.php" style="text-decoration: none">Live <sup>in</sup> Care</a></h2>
    <nav>
        <li><a href="index.php" style="text-decoration: none">Kezdőlap</a></li>
        <li><a id="childrenbtn" href="childcheck.php" style="text-decoration: none"> Gyerek követés</a></li>
        <li><a id="hchecker" href="healthschecker.php" style="text-decoration: none"> Ellátás</a></li>
        <li><a href="logout.php" style="text-decoration: none"> Kilépés</a></li>
    </nav>
</header>
```

XXIX. ábra: childchecker.html

A kezdő lap az homepage2.html nevű fájl van benne. Itt a body része van benne, meg a footer és html záró része (a bodynak és a html-nek záró tagje). Kezdő lapom <section>-kből épül fel, azért úgy csináltam, hogy egyes menü pontokból legördüljön adott web részhez.

Főbb részei:

- Főképernyő (betöltéskor ezt láttod betöltéskor), Rólunk, Szöveg mögötti kép, Idézetek, Szöveg mögötti kép, Footer rész

Rólunk és Idézeteket egy sima mentén csináltam meg. *XXX. ábrán* látható a szerkezeti felépítése. A class=company-left -nél található a kép css-sel oldottam meg a kép beillesztést és a bal felére helyezést (felosztást). A másik oldalon van a szöveg class=company-right -nál, ami viszont jobbra van helyezve. Jobbra és balra helyezést felosztás segítségével csináltam meg, ami nem más, mint flex-basis, és meg kell adni mennyi százaléka legyen az egyik oldal és a másik oldal a szélességből. Valamint egy gombot helyeztem el, ami egy url-re mutat, és nyit meg kattintáskor. Rólunk címet is css-be formáztam.


```
.company-title
{
  text-transform: uppercase;
  font-size: 50px;
  margin-bottom: 5%;
}
```

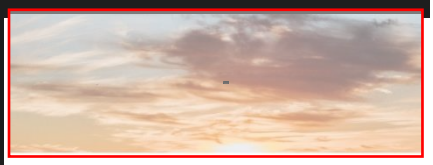
```
.company-left
{
  flex-basis: 40%;
  background-image: url("../Pics/company.jpg");
}
```

```
<!--------------------------------Cégről form-------------------------------->
<section class="campany-area" id="company">
    <h3 class="second" class="company-title">Rólunk</h3>
    <ul class="company-content">
        <li class="company-left"></li>
        <li class="company-right">
            <h2>A cégünkről</h2>
            <p>A Live <sup>in</sup> Care által fenntartott hálózatokban és ifjúság házakban ma közel 400 gyermek és fiatal él.
            Az önállóság következő lépcsőfokában a Live <sup>in</sup> Care által fenntartott lakásba költözhetnek, ahol már önál
            <br><br>
            <a id="info" style="text-decoration: none" href="https://www.sos-childrensvillages.org/"> &darr;</a>
            <br>
        </li>
    </ul>
</section>
```

XXX. ábra: homepage2.html-ből Rólunk magyarázat (style.css /lent/, homepage2.html /fent/)

Háttérben a képeket css segítségével oldottam meg. Középre helyeztem a background-position:center-rel, majd a background-size: cover-rel a hattérbe helyeztem, majd fixaltam hogy ott nem mozduljon el a background-attachment: fixed -del. *XXXI. ábra* látható piros keretbe ezt a terület.

```
<!--------------------------------Felső kép---------------------------------->
<section id="midpic">
</section>
#midpic
{
  background-image: url(../Pics/midpic.jpg);
  background-position:center;
  background-size: cover;
  background-attachment: fixed;
  padding:14em;
}
```



XXXI. ábra: Szövegek mögötti kép (style.css, homepage2.html)

Idézetek

e. Regisztrációs felhasználónév és jelszó használata:

```
<!--------------------------------Regisztráció teszt-------------------------------->
    <div id="id02" class="modal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
      <form class="modal-content animate" method="POST">
        <div class="imgcontainer">
          <span onclick="document.getElementById('id02').style.display='none'" class="close" title="Close Modal">&times;</span>
        </div>

        <div class="container">
          <label><b>Regisztrációsnév:</b></label>
          <input id="runame" type="text" placeholder="Registration username" name="reguname" required>

          <label><b>Regisztrációs jelszó::</b></label>
          <input id="rrpsw" type="password" placeholder="Password" name="psw" required>
          <p></p>
          <button id="rloginbtn" type="submit">Regisztráció</button>
        </div>
        <button type="button" onclick="document.getElementById('id02').style.display='none'" class="cancelbtn">Cancel</button>
      </form>
    </div>
```

XXXII. ábra: Regisztrációs jelszó ellenőrzése (homepage2.html)

A webes felületre kizárólag csak szülők tudnak bejelentkezni. Amikor egy szülőt beregisztrálnak az asztali alkalmazáson, akkor kap a nevéből generált regisztrációs nevet és ehhez egy regisztrációs jelszót, amit majd meg kell változtatnia.

```
if (!empty($_POST['rusername']) && (!empty($_POST['rpassword'])) && ($_SERVER["REQUEST_METHOD"]=="POST")) {
    $username = $_POST['rusername'];
    $pwd = $_POST['rpassword'];
    $sql = "SELECT * FROM parents WHERE loginuser = '$username' AND loginpsw = '$pwd' AND loginpermission =1 AND reg=0";
    $res = $conn -> query($sql);
    if (!$res){
        die("Hibás bejelentkezés!");
    }
    if ($res -> num_rows == 1){
        //sikeres a belépés
        $row = $res -> fetch_assoc();
        $_SESSION['azonosito'] = $row['ID'];
        echo "1";
    } else
    {
        //sikertelen bejelentkezési kísérlet
        echo "0";
    }

}
```

XXXIII. ábra: Regisztrációs jelszó ellenőrzése (reg.php /fent/, login.js /lent/)

```
$("#rloginbtn").click(function(e){
    e.preventDefault();
    let rusername = $("#runame").val();
    let rpassword = $("#rrpsw").val();
    //console.log(rusername);
    //console.log(rpassword);
    $.ajax({
        url:"reg.php",
        method:"POST",
        data:{rusername:rusername,rpassword:rpassword},
        success:function(data){
            //alert(data);
            if(data == "1")
            {
                $("#id02").fadeOut(300);
                $("#id03").fadeIn(300);
                $.ajax({
                    url:"preg.php",
                    method: "POST",
                    data:{rusername:rusername,rpassword:rpassword},
                    success: function(data){
                        $(".reger").html(data);
                    }
                });
            }
```

A login.js (kék keretes kép, *XXXIII. ábra*) fájlban a gomb lenyomásakor, kiolvassa a két inputból a beírt felhasználónevet és jelszót, ezt elmenti, majd át adja reg.php-nak, az le ellenőrzi, hogy van -e ilyen jelszó pár az adatbázisban, és regisztrált -e már (adatbázisban ez a reg oszlop, ha igen akkor 1-es van) és van -e engedélye a regisztrációra (loginpermissio az adatbázisban, ha igen akkor 1). Ha igen, akkor login.js-ben, data egyenlő lesz 1-gyel, és ez a modal (*XXXII. ábra*) ablak eltűnik és felugrik egy másik. Utána megnyitja a második modult, ha helyes a regisztrációs jelszó és felhasználónév (*XXIV. ábra*).

f. Adatain ellenőrzése és jelszó változtatás:

```
<!-------------------------------Regisztráció ------------------------------->
<div id="id03" class="modal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <form class="modal-content animate" method="POST">
    <div class="imgcontainer">
      <span onclick="document.getElementById('id03').style.display='none'" class="close" title="Close Modal">&times;</span>
    </div>

    <div class="container reger" >

    </div>
    <button type="button" onclick="document.getElementById('id03').style.display='none'" class="cancelbtn">Cancel</button>
  </form>
</div>
```

XXXV. ábra: Adatok mentése, jelszó változtatása (preg.php)

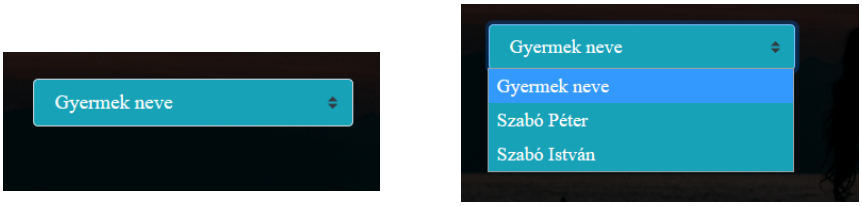
```
if (!empty($_POST['rusername']) && (!empty($_POST['rpassword'])) && ($_SERVER["REQUEST_METHOD"]=="POST")) {
    $username = $_POST['rusername'];
    $pwd = $_POST['rpassword'];
    $sql = "SELECT * FROM parents WHERE loginuser = '$username' AND loginpsw = '$pwd' AND loginpermission = 1";
    $res = $conn -> query($sql);
    $html = "";
    while($row = $res->fetch_assoc())
    {
        $html .='
        <label><b>Név:</b></label>
        <input id="regname" type="text" value="'.$row['pname'].'" name="reguname" required>

        <label><b>Születési idő:</b></label>
        <input id="regbrth" type="text" value="'.$row['pbirth'].'"  name="psw" required>

        <label><b>Személyigazolványszám:</b></label>
        <input id="regidcard" type="text" value="'.$row['pidcardnumber'].'"  name="psw" required>

        <br>
        <br>
        Adja meg az új jelszavát és felhasználó nevét:
        <br>

        <label><b>Felhasználónév:</b></label>
        <input id="regusername" type="text"  name="reguname" required>

        <label><b>Jelszó:</b></label>
        <input id="regpassword" type="password"  name="rpsw" required>

        <button class="save">Mentés</button>
        ';
    }
    echo $html;
```

XXXIV. ábra: Regisztráció mentés (homepage2.html)

XXXV. ábrán az látható, hogy *XXXIV. ábrán* reger class nevű div-et feltölti az adott felhasználó adataival, amit preg.php végez el, az ajax által átadott jelszó és felhasználónév segítségével. Miután az új jelszót és felhasználó nevet beírtuk (*Felhasználónév betű és számból állhat minimum 6 karakter. A jelszó pedig számból és betűből kell állnia, de minimum 8 karakter. RegEx-szel ellenőrzöm ezt.*), és rákattintunk a mentés gombra, akkor az adatbázisban reg oszlopban a cellát 1-re álltja ennél a felhasználónál. Ami azt jelenti, hogy regisztrált felhasználó. Ezután már nem tud regisztrálni újból. Emellett természetesen a jelszót is megváltoztatja az új jelszóra és a többi változtatott adatot is, ha azok nem voltak megfelelően félvéve. Mert asztali alkalmazás által helyesen lettek félvéve, de itt van lehetőség az adatokat

ellenőrizni és esetlegesen javítani a rosszul felvitt adatot. Utána már lehetséges is a bejelentkezés. Bejelentkezést login.php ellenőrzi, amit login.js ad át neki. Ugyan az a menete, mint a *XXXIII. ábránál*. Ellenőrzés és ha van egyezés az adatbázisban, akkor csak itt átirányít az index.php-ra a bejelentkezés után. Itt játszik szerepet menuin.html, menu.html és a childchecker.html, mert megváltozik a menü sáv. Például a Regisztrációs menü pont eltűnik bejelentkezéskor hiszen, akkor már nincs szükség rá.

Miután beléptünk két új menüpont lesz. Gyerek követés és Ellátás menüpontok. Két rész majdnem ugyan úgy működik csak más adatokkal. Gyerek követés szemszögéből magyarázom el a működését. Gyerek követés, azt a funkciót adja, hogy adott eseményt kiírja, ami a gyerekkel történt, a kiválasztott név alapján (Kirándulás a mecsekbe). Ellátásnál, pedig az egészségügyi vizsgálatokat, amit a gyereken végeztek (szemvizsgálat). Miután rákattintasz a Gyerek követés menüpontra, akkor átirányít a gomb childcheck.php-ra.



XXXVI. ábra: Legördülő menü

```php
if($_SERVER['REQUEST_METHOD']=="POST" && !empty($_POST['names']))
{

    $childn=$_POST['names'];
    $sql ="SELECT children_healths.type,children_healths.details,children_healths.special_treatment,children_healths.treatdate,children_healths.by
    FROM children_fullprofile INNER JOIN children_healths ON children_fullprofile.ID = children_healths.childrenID
    WHERE children_healths.childrenID= '$childn'";
    $res = $conn ->query($sql);
    $html = "
    <thead class='bg-info text-white'>
        <tr>
            <th>Tipusa</th>
            <th>Leírás</th>
            <th>Különleges bánásmód</th>
            <th>Vizsgálat dátuma</th>
            <th>Orvos</th>
        </tr>
    </thead><tbody class='text-info'>";
    while($row = $res -> fetch_array())
    {
        $html.=
        "<tr>
            <td>{$row[0]}</td>
            <td>{$row[1]}</td>
            <td>{$row[2]}</td>
            <td>{$row[3]}</td>
            <td>{$row[4]}</td>
        </tr>";
    }
    $html .= "</tbody>";

    echo $html;
```

```php
<section id="up" class="start-area">
  <div class="start-img"></div>
  <h1 class="slower animated bounceInLeft">Live <sup>in</sup> Care gyerek követő</h1>
  <h1 class="slower animated bounceInLeft">Válassza ki a gyermek nevét a legördülő menüből!</h1>
  <div class="container">
    <div class="row">
      <div class="col-lg-3">
        <?php
          $parentId = $_SESSION['uid'];
          $sql="SELECT DISTINCT children_fullprofile.cname,children_fullprofile.ID
          FROM children_fullprofile INNER JOIN ceventsk ON children_fullprofile.ID = ceventsk.childrenID INNER JOIN children_events ON
          children_events.ID = ceventsk.eventsID INNER JOIN parentsk ON children_fullprofile.ID = parentsk.childrenID
          WHERE parentsk.pID = '$parentId'";
          $res = $conn -> query($sql);
          $select = "<br><select class='custom-select bg-info text-white' id='chiname'><option>Gyermek neve</option>";
          while ($row = $res->fetch_array())
          {
            $select .= "<option id='$row[1]'>{$row[0]}</option>";
            $choseone = $row[0];
          }
          $select .= "</select><br><br>";
          echo $select;
        ?>
      </div>
    </div>
    <div class="row">
      <div class="col-lg-12">

      <table class="chitable table table-light table-borderless table-striped">
      </table>
      </div>
    </div>
  </div>
</section>
```

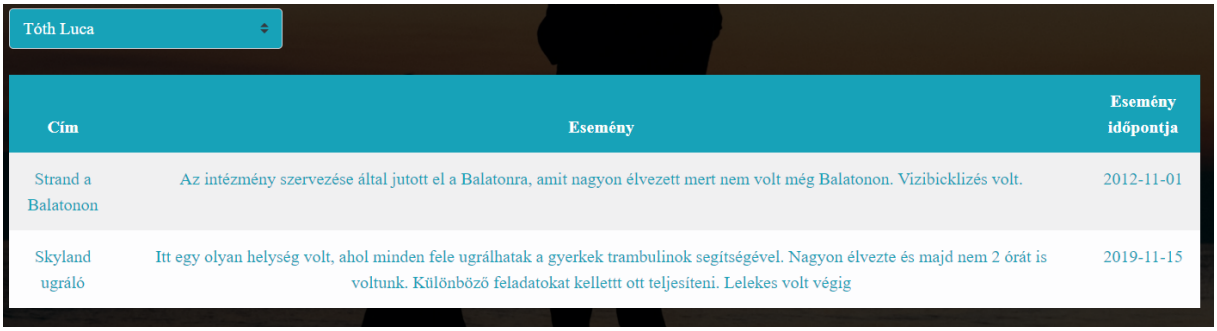
XXXVII. ábra: Gyerek eseményeit kiírása, kiválasztott gyerek alapján (childchecker.php)

Itt a legördülő menü-t (*XXXVI. ábra*) feltölti az adott szülőhöz tartozó gyerekek neveivel, de csak egyszer az $sql segítségével, ha tartozik a gyerekéhez bejegyzés. Miután kiválasztunk egy gyereket a legördülő menüből, a kiválasztott névhez tartozó eseményeket megjelenít egy táblázatban. Ezt a következő képen csinálja.

```
$("#chiname").change(function(){
    let names = $(this).children(":selected").attr("id");
    //console.log(names);
    $.ajax({
        url:"selectchildrendata.php",
        method:"POST",
        data:{names:names},
        success:function(data){
            //alert(data);
            $('.chitable').html(data);
        }

    });
});
```

XXXVIII. ábra: Gyerek eseményeinek kiírása (login.js, selctchildrendata.php)

Miután kiválasztottuk egy gyerek nevet, login.js ezt tovább adja a névhez tartozó id-t selectchildrendata.php-nak, ezt id-t bele rakja $sql változóba és leszűri az adott névhez tartozó eseményekre és csak azokat kéri le. Ha van egyezés



XXXIX. ábra: Gyerek eseményei a weblapon

az ajax beilleszti az html kódot a chitable nevű classba (ebben az esetben table-be). A kiírásra egy példa *XXXIX. ábra*. Ha nincs ilyen esemény, akkor a gyermek neve meg se jelenik a legördülő menübe (csak „Gyermek neve" lesz a legördülő menübe), és ki fogja írni, hogy „Nincs a gyerekéhez/gyermekeihez esemény hozzárendelve". Ugyan ilyen módon működik az Ellátás is, csak más adatokkal.

g. Fájlok, szerkezeti felépítés

- HTML
  - childchecker.htm (menü sáv változtatásra szolgál, *XXIX. ábra*), header.html (*XXIV. ábra*, hivatkozások), homepage2.html (homepage), menuin.html (menü sáv változtatásra szolgál, *XXVIII. ábra*), menu.html (menü sáv változtatásra szolgál, *XXVII. ábra*)
- JS
  - login.js (ajax, ellenőrzés), jquery-3.4.1.min.js, bootstrap.min.js
- CSS
  - style.css (formázások), animate.css (a kezdő képernyőn a szöveg meganimálása), bootstrap.min.css (boostrap formázások), font-awesome.min.css (betű formázás)
- PHP
  - childcheck.php, healths.php, healthschecker.php, index.php, login.php,
  - logout.php, preg.php, reg.php, selectchildrendata.php, selectchildrendatah.php, updatepdata.php, connect.php, functions.php, init.php

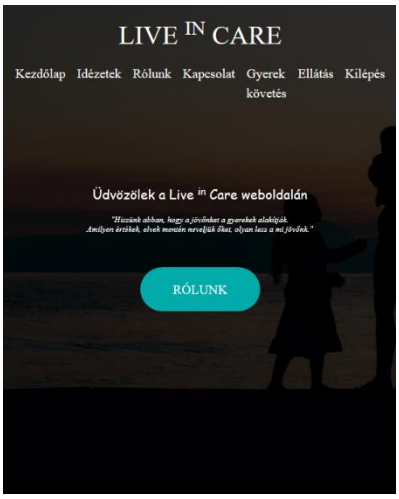
h. Reszponzivitás

A weblapom, hogy reszponzív legyen azt css-ben oldottam meg nagyobb részben (*XXXIX.I. ábra*). A modal ablakot és aztáblázatba kiírt információkat Gyerek követés és Ellátás menüpontban azokat viszont boostrap segítségével.

```css
/*kisebb felbontásnál összébb tólás 1000px*/
@media(max-width:1000px)
{
  section
  {
    padding: 100px 50px;
  }
}
/*távolság szerkeztés 600px*/
@media(max-width:600px)
{
  section
  {
    padding: 125px 30px;
  }
}
```

```css
@media (max-width:800px)
{
  .start-area
  {
    min-height: 600px;
  }
  .start-area h1
  {
    font-size: 16px;
  }
  .start-area h3
  {
    font-size: 10px;
  }
   .start-area a.start-btn
  {
    padding: 15px 40px;
  }
}
```

XXXIX.I. ábra: A kezdőlap reszponzív beállításai



XXXIX.II. ábra: A kezdőlap

i. Fejlesztési lehetőség:

A weblapomról egy két funkció szükséges lenne, de az idő szűkösségé miatt nem tudtam hozzáadni a weblapomhoz. De a következő hónapokban ezt hozzá szeretném tenni.

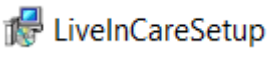
- A felhasználó tudjon jelszót változtatni, visszajelzést tudjon adni, vagy
hibaüzenetet, saját profil megtekintés

4. Felhasználói dokumentáió:

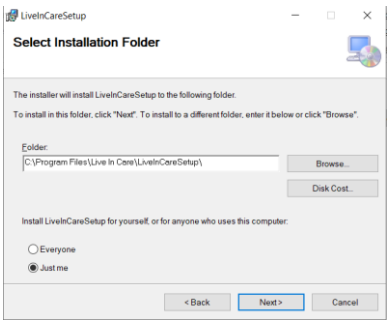
I. Szoftver

a. Telepítés

Az alkalmazás telepítő fájlát a gépünkre másoljuk, majd dupla kattintással elindítjuk. Majd a Next gombra kattintva, a következőt látjuk *XXXX. ábra*.

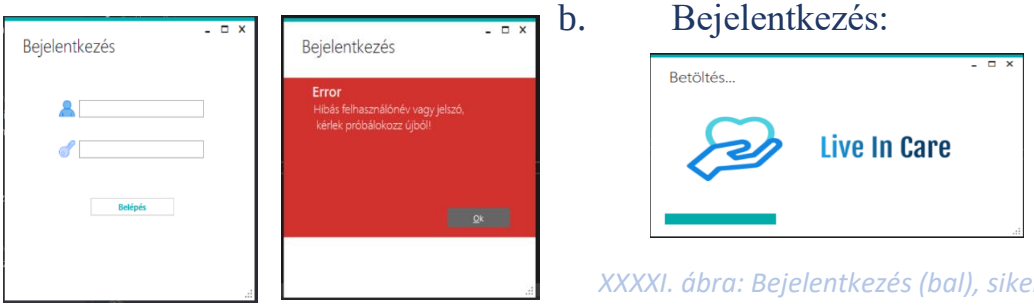


XXXX. ábra: A telepítő fájl balra, és jobbra a telepítése



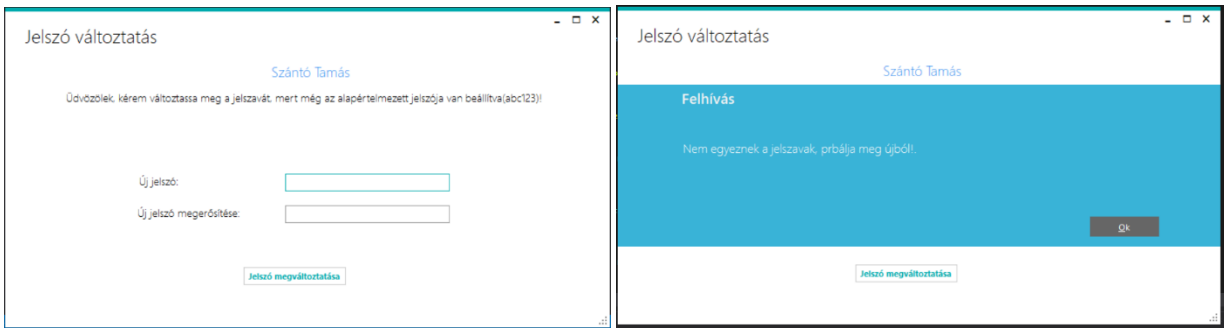
A Folder: alatti részben írja azt, hogy hova telepíti automatikusan a programot a telepítő. Ez jelenesetben ez: C:\Program File\Live In Care\LiveInCareSetup mappában lesz. Ha ezt meg szeretnénk változtatni, a Browse… -ra kattintva megváltoztathatjuk a telepítés helyét. Ekkor felugrik egy ablak és ott kell kiválasztanunk az adott mappát, ahova szeretnék telepíteni. Ezek után 2-szer a Next gomb és fel is van telepítve a program a gépünkre.

b. Bejelentkezés:



XXXXI. ábra: Bejelentkezés (bal), sikertelen bejelentkezés (közép), Töltés képernyő (jobbra)

A program elindításakor a Töltés képernyő indul el, amikor fontosabb műveleteket hajt végre (adatbázis létrehozás). Miután ez sikeres volt, megjelenik a Bejelentkezési felület. Itt két szöveges mező található. Az elsőbe kell beírni a felhasználó nevet és a másodikba a jelszót. Fontos, hogy figyeljen oda a kis és nagy betűre, mert az különbözőnek számít. De erre figyelmeztet a bejelentkezés közben, hogy Caps Lock be lett nyomva. Ha a felhasználó név és jelszó pár helytelen, akkor egy hiba üzenetet ír ki (*XXXXI. ábra*). Ilyenkor az Ok gombra kell kattintani, majd újra beírni a jelszót, csak helyesen. Miután ez sikeres volt a bejelentkezés, a munkahelyén betöltött pozíció alapján betölti a felültet. Ha első bejelentkező, akkor egy jelszót kell változtatnia, mert alapértelmezett jelszó nem maradthat.

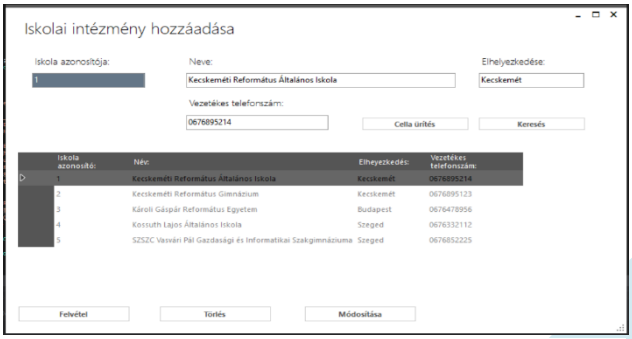


XXXXII. ábra: jelszó változtatási felület.

XXXXII. ábrán a jelszó változtatás látható. Ha alapértelmezett jelszót írtál be, akkor kér a program, hogy ezt változtasd meg. Kétszer kell beírnod a jelszót, és akkor fogadja el, ha két ugyan olyan jelszót adtál meg. Hanem akkor figyelmeztet erre a program (*XXXXII. ábra*). Bármilyen jelszót meg lehet adni. Nincs megkötés. Majd a Jelszó változtatás gombra kattintva végbe is megy.

c. Felvétel, módosítás, törlés, keresés

3 féle műveletet lehet csinálni a regisztrációs felületeken. Ezek a felvétel, módosítás és törlés. Felvétel során a bevitt adatoknak meg kell felelni a beviteli szabályoknak, nem lehet üres. Ha karakter számot átlépi, akkor adatvesztés lehetséges. A program erre nem figyelmeztet. De azok szöveg mezők, amik be vannak színezve egy adott felület színével, azok nem



XXXXIII. ábra: Sor kijelölés

módosíthatok. Például a *XXXXIII. ábrán* az Iskola azonosító-ja mert az automatikusan generálódik. Ezeket a Pozíció szerinti felülettek-nél fogom elmagyarázni egyes regisztrációknál, mire kell figyelni. Módosítás és törlésnél fontos, hogy egy adott sora ki legyen jelölve mielőtt, töröl vagy módosít. Keresésnél egy adott cella segítségével lehet keresni, és utána keresés gombra kattintással. Minden felületnél majd, hogy nem más. Ezt is a Pozíció szerinti felülettek-nél fogom leírni, melyik cella segítségével lehet keresni.

d. Pozíció szerinti felülettek:

5 féle bejelentkezési pozíció van.

- Intézményvezető, Ügyintéző, Nevelő, Pszichológus, Admin / Rendszergazda

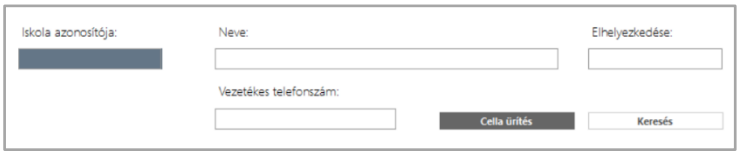
A következőkben sora fogom venni mindegyiket, és megemlítem mire kell oda figyelni. Mindegyik felületen van lehetőség kijelentkezni (Kijelentkezés) és utána bejelentkezni akár egy másik felhasználóval. A másik lehetőség a hiba bejelentés, ahol az elérhetőség jelenik, akit lehet ezzel kapcsolatban hívni és email-t írni neki. Minden felület felső sarkában lehet olvasni az adott bejelentkező nevét (*XXXXIII. ábra*).

- Nevelő:

Miután beléptél nevelőként a programon keresztül ez a felület jelenik meg előtted. Hamarosan elérhető felirat azt jelenti, hogy a következő frissítésnél ez elérhető lehet majd. Ezen a felületen van egy különleges rész az, hogy a pszichológus vizsgálatokat a nevelő megnézi, de csak megtekintheti, de nem módosíthat és nem törölni.

XXXXIV. ábra: Nevelő felülete

Nevelő oktatás intézményt tud felvenni, és majd gyereket hozzárendelni. Valamint eseményt felvenni, majd ezt is gyerekhez rendelni.

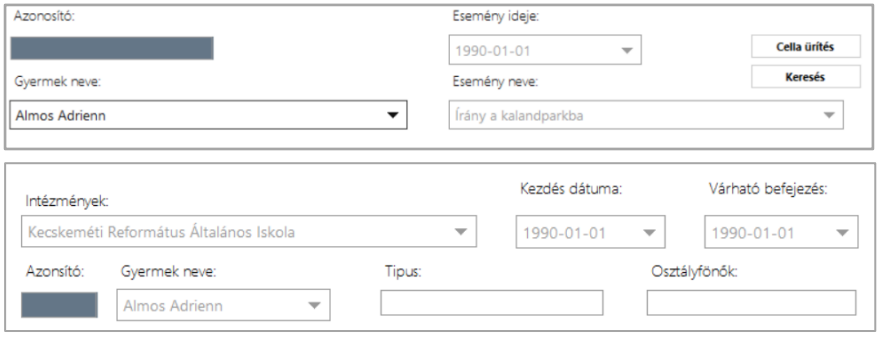


XXXXV. ábra: Oktatási intézmény regisztrációja

Oktatási intézmény Neve nagy betűvel kell kezdeni, és maximum 80 karakter hosszú lehet. Vezetékes telefonszámnál ezt a formátumot kell követni: 0676123123. Vagyis 0676-tal kell kezdeni és utána csak hat szám írható. Elhelyezkedésnél város nevét kell beírni nagy betűvel és maximum 40 karakter. Keresés pontos Intézmény Neve alapján lehet.

Esemény regisztrációjánál, a Cím-t nagy betűvel kell kezdeni és maximum 30 karakter hosszú. Leírás 255 karakter hosszú lehet.

XXXXVI. ábra: Esemény regisztrációja



XXXXVII. ábra: Esemény, oktatási intézmény gyerekhez kapcsolása

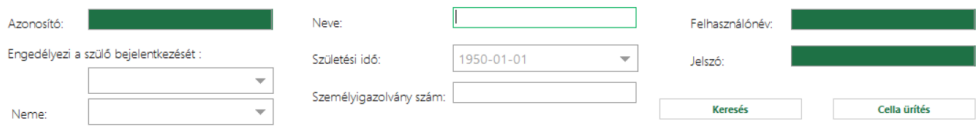
Az esemény és oktatási intézménynél arra kell figyelni, hogy a legördülő menüből legyen egy név kiválasztva. Az eseménynél Gyermek neve és Esemény neve, az oktatási intézménynél Gyermek neve, és Intézmény -nél kell egyet kiválasztani. Fontos, hogy itt a legördülő menüben csak akkor jelenik meg az intézmény neve, vagy esemény címe, ha az előtte regisztrálva lett (*XXXXV. ábra, XXXXVI. ábra*). Dátumot pedig alaphelyzetből [14]át kell állítani. Ide tartozik Esemény ideje, Kezdés dátuma, Várható befejezés. Itt is a legördülő menüből lehet adott dátumot kiválasztani. Keresés az esemény-gyerek párnál a Gyerek neve alapján lehet. Az oktatás-gyermek egymáshoz rendelésnél pedig Iskola név alapján lehet.

- Ügyintéző:

A gyerek regisztrációja nagyon fontos, hiszen több regisztrációnál is használja a gyerek nevét. Minden adat helyesen legyen megadva.

XXXXVIII. ábra: Esemény, oktatási intézmény gyerekhez kapcsolása

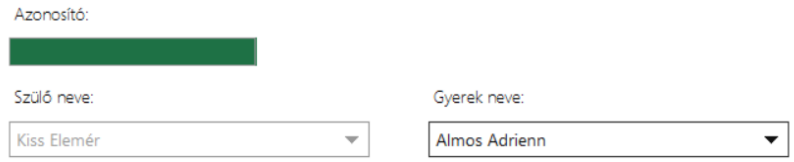
A Neme -ét ki kell választani a legördülő menüből, maximum 50 karakter hosszú. Neve csak nagy betű kezdéssel érvényes. Személyigazolványszám összesen 8 hosszú és első 6db szám aztán 2 db nagy betű követi (123123AV). TAJ szám pedig 9 db számból áll. Születési időt meg kell változtatni, Születési hely nagy betűvel kell kezdeni, és maximum 40 karakter hosszú. Belépés időpont-ját meg kell változtatni és Intézmény helye, ahol tartózkodik az pedig nagy betűvel kell kezdődjön, maximum 40 karakter hosszú. Keresés Neve alapján lehet.



XXXXIX. ábra: Szülő regisztrációja

Engedélyezi a szülő bejelentkezést, ez különösen fontos hiszen ez határozza meg hogy egy szülő beléphet egy webes felületen. Kikkel választani Engedélyezés vagy Tiltást. Nemét ki kell választani, Nevét meg kell adni nagy kezdő betűvel, maximum 40 karakter hosszú lehet. Születési időt meg kell változtatni, Személyigazolványszám 8 hosszúságú, 6db szám 2 db nagy betű, ebben a sorendbe (123123AS). Ezeket jól kell felvinni mert bejelentkezéskor ezt a szülő ellenőrzi, hogy jól lettek -e felvéve. Keresés Neve alapján lehet. A felhasználó név és jelszó automatikusan generálódik, hozzáadás utána a táblázatba látszik mi lett a jelszó és felhasználónév.

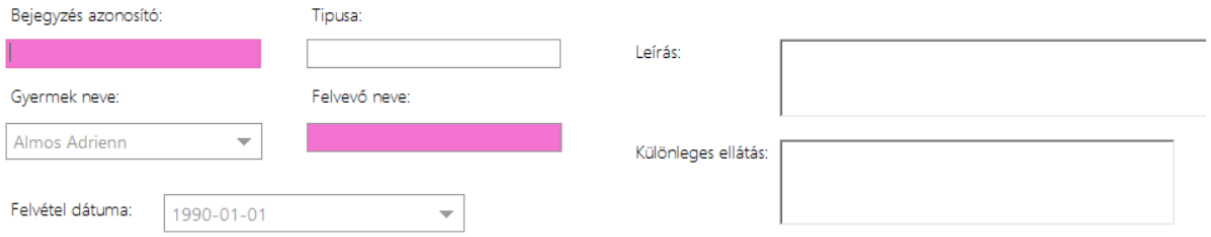
[14] Azt jelenti, hogy a betöltéskor mutatott dátumtól különbözőt kell kiválasztani.



XXXXX. ábra: Szülő gyerekhez rendelésének regisztrációja

Nagyon egyszerű, csak ki kell a gyereket választani és a hozzá tartozó szülőt. Gyerek neve alapján lehet keresni.

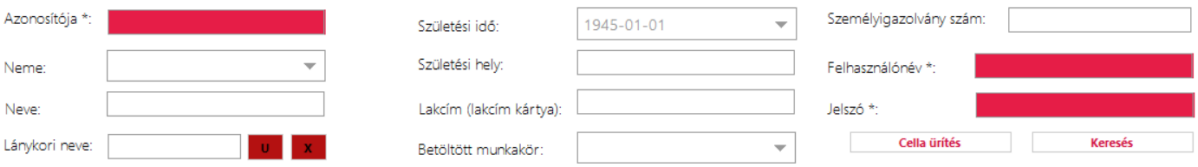
- Pszichológus:



XXXXXI. ábra: Gyerekkel történt vizsgálatok regisztrációja

A Gyermek nevét kell kiválasztani. Fontos, hogy itt csak a beregisztrált gyerekek nevei vannak. Típusa, oda a vizsgálatot típusát kell írni például Szemvizsgálat, nagy betűvel kell kezdeni és maximum 20 karakter hosszú lehet. Leírás oda az fog menni, hogy pontosan mi volt és miért kellet elvégezni a vizsgálatot, nagy betűvel kell kezdeni és maximum 255 karakter hosszú lehet. Különleges bánásmódhoz azt kell írni, hogy milyen oda figyelést kell feltétlen csinálunk a gyerekkel szemben. Például Szemvizsgálatnál, hogy hordja a szemüveget. Felvétel dátumához a vizsgálat időpontja kell, amikor történt és meg kell változtatni.

- Intézményvezető:



XXXXXII. ábra: Dolgozók regisztrációja

Dolgozó regisztrációja a feladata. Neménél ki kell választani a nemét, Nevét nagy betűvel kell kezdeni, maximum 50 karakter hosszú. Lánykori neve, mellette van egy x és u betű. X azt jelöli, hogy nincs neki, tehát férfi. U pedig azt jelöli, hogy ugyan az, mint a rendes neve például nem házas nő, de ekkor maximum 50 karakter. Születési időt meg kell változtatni, Születési helyét nagy betűvel kell kezdeni és maximum 40 karakter. Lakcíme mezőt nagy betűvel kell kezdeni, és számot is kell tartalmazni, a házszám miatt és maximum 90 karakter. Személyigazolványszámnak 8 hosszú lehet, 6db szám és 2db nagy betű ebben a sorendbe. Betöltött munkakörnél ki kell választani a megfelelőt. A felhasználónév és jelszó automatikusan generálódik, az alatta lévő táblázatba látszik hozzáadás után.

e. Hiba jelzés

XXXXXIII. ábra: Hibás adatbevitel jelzése

A felvétel, vagy módosítás során nem megfelelően beírt vagy üresen hagyott cellát így jelzi, hogy nem megfelelő a bemeneti adat. Villogva jelenik meg, majd a kurzort rárak ki is írja a hiba okát (*XXXXIII. ábra*).

II. Weblap

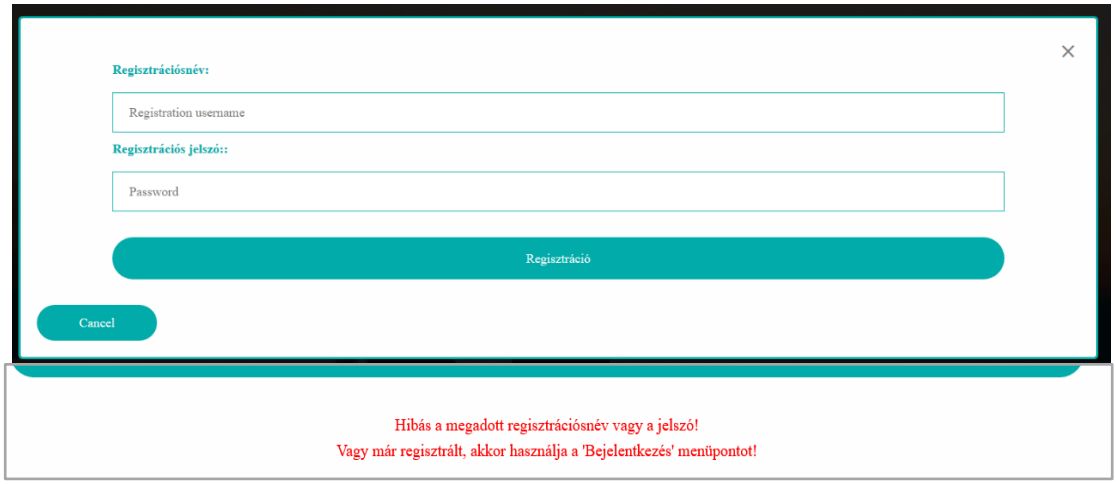
XXXXXIV. ábra: Menü

a. Kezdő lépernyő részei:

A weboldal betöltésekor a menü 6 menüpont van (*XXXXXVI. ábra*). Kezdőlap, amire ha rákattintunk akkor a kezdőképernyőre juthatunk el, majd hogy nem egy lap frissítés. Regisztráció és Bejelentkezés, amiket későbbi pontokban kifejtek. Idézeteknél, híresebb emberek idézeteit olvashatjuk el, akik a gyerekek kapcsolatban mondtak. Ami motiváció is lehet az ilyen szférában dolgozó embereknek. Rólunk menü pontba a céget foglalja összeröviden, és nemzetközi céget lehet elérni onnan a kék nyílra kattintva. Kapcsolat menüpont kattintása során pedig, a cég elérhetőségét láthatjuk.

b. Bejelentkezés

Ahhoz, hogy be tudjon jelentkezni, regisztrálnod kell. Regisztrációs kódot és felhasználó nevet az intézménytől lehet elkérni, mert asztal alkalmazás szülő regisztrációnál generálódik felvétel közben. Fontos része, hogy csak akkor tud egy felhasználó regisztrálni a bejelentkezéshez, ha erre engedélyt kapott, vagyis asztali alkalmazáson úgy lett felregisztrálva. Miután meg van a regisztrációs jelszó és hozzá tartozó felhasználónév is, akkor válaszuk ki a Regisztráció menüpontot.



XXXXXV. ábra: Regisztrációs form (fent), hiba üzenet (alul)

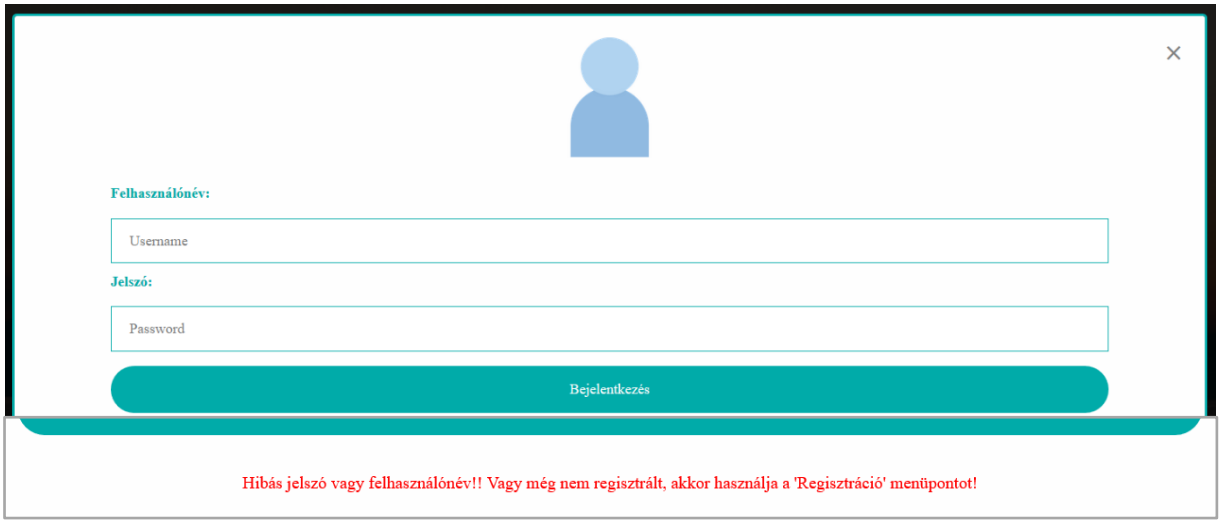
Első szöveg dobozba kell írni a felhasználó nevet és a második a jelszót. Ha már a felhasználó regisztrált, akkor az alsó hiba üzenetet írja ki, vagy ha nincs engedélye a regisztrációra vagy ha rossz regisztrációs jelszó, felhasználónév párt írt be. Cancel gombbal vissza tud lépni a

kezdő képernyőre. Ha sikeres a bejelentkezés, akkor egy adat ellenőrzést kép a jelszó változtatással együtt, mert meg kell változtatni az alapértelmezett jelszót!

c. Adatok ellenőrzése regisztráció után:

XXXXXVI. ábra: Regisztrációs adatok ellenőrzése, jelszó változtatás

Ezen a felületen nagyon alaposan le kell ellenőrizni a személyes adatokat, hogy helyesen lettek -e felvéve. Ha igen, akkor azzal nincs más teendő. Ha nem akkor az adott adatot át kell írni a valósra. Utána meg kell adni egy új jelszó és új felhasználó nevet. Fontos, hogy a felhasználó név számot betűt tartalmazhat, de minimum 6 karakter. Még a jelszó betűt és számot is kell tartalmazni, de minimum 8 karakter. Érdemes felírni az új jelszót és felhasználó nevet. Ezek után rá lehet kattintani a Mentés gombra. Utána az új jelszó és felhasználóval be is lehet jelentkezni a Bejelentkezés menüre kattintva (*XXXXXIV. ábra*).



XXXXXVII. ábra: Bejelentkezés felülete (felül), hibaüzenet (alul)

Bejelentkezés menüpontra kattintva beléphetünk, helyes jelszó és felhasználónév használatával.

d. Gyerek követés, Ellátás menü pontok

Három új menüpont jelenik meg a Gyerek követés, Ellátás és a Kilépés bejelentkezés alkalmával. Kilépés gombbal ki tudunk jelentkezni a fiókunkból. Gyerek követésnél a

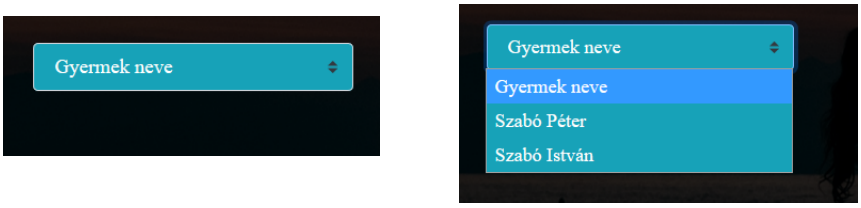
gyerekkel történt eseményeket tekinthetjük meg. Míg az Ellátás menüpontban az egészségügyi vizsgálatokat, amin a gyerek részt vett.

XXXXXVIII. ábra: Az új menüpontok

Mind két menü pont ugyan úgy működik, szóval az egyiken keresztül bemutatom. Ha rákattintunk Gyerek követés menüpontra, akkor betölt egy weboldalt.

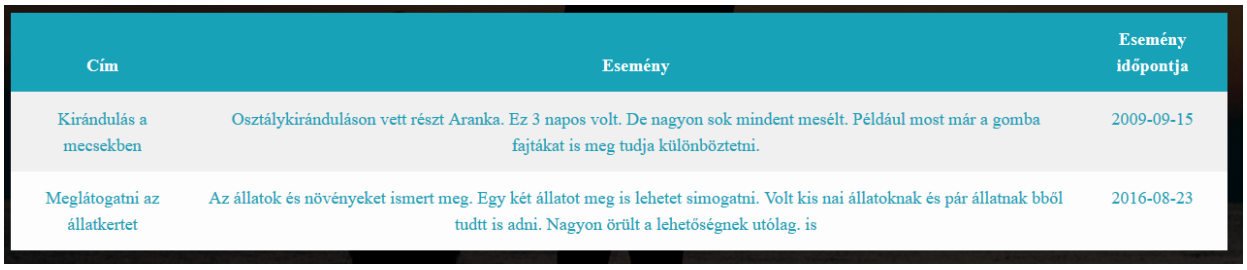
XXXXXIX. ábra: Gyerek követés menü

A betöltött weboldal közepén XXXXXIX. ábra látható. Itt a legördülő menüből ki tudjuk választani a gyerek nevét. Ha több gyerekünk is van a rendszerbe, akkor mindegyik neve bene lesz.



XXXXXIX. ábra: Gyerek követés menü

A kiválasztott név után egy táblában írja ki ő hozzá tartozó eseményeket, a következő képén:



Cím	Esemény	Esemény időpontja
Kirándulás a mecsekben	Osztálykiránduláson vett részt Aranka. Ez 3 napos volt. De nagyon sok mindent mesélt. Például most már a gomba fajtákat is meg tudja különböztetni.	2009-09-15
Meglátogatni az állatkertet	Az állatok és növényeket ismert meg. Egy két állatot meg is lehetet simogatni. Volt kis nai állatoknak és pár állatnak bből tudtt is adni. Nagyon örült a lehetőségnek utólag. is	2016-08-23

XXXXXX. ábra: Gyerek esemény kiírás táblázatban

Ugyan így van az ellátás menüponttal is csak itt az egészségügyi vizsgálatokat írja ki, ami a kiválasztott névhez tartozik:

Tipusa	Leírás	Különleges bánásmód	Vizsgálat dátuma	Orvos
SNI	Nehézen lehet a figyelmét lekötni, inkább a saját útját járja. Ezzel rendszeresen zavarva a társait. Tanulás folyamata sokkal lassan mit vele egy körű társaihoz hasonlítva. Ezért kivonja magát ebből.	Speciális oda figyelés, figyelem fejlesztés.	2008-08-15	Major Anna

XXXXXXI. ábra: Gyerek egészségügyi vizsgálat kiírás táblázatban

5. Ergonómia

A web lapomat és a asztali alkalmazásomat is úgy alakítottam ki hogy színek alapján könnyen be lehessen azonosítani, de emellett ne legyen sok belőle, hogy ne tűnjön bonyolultnak. Egyes hibajelzéseket is különböző színekkel láttam el, hogy könnyen felismerhető legyen a közlés és magyarázó hangnembe van. Egyes nem megfelelő műveleteknél is, a felhasználóval közli a probléma megoldását.
Asztali alkalmazáson a bejelentkezéstől függően változik a felületek színe. Míg a weboldalon úgy állítottam be, hogy asztali alkalmazás alap színe legyen. Például az asztali bejelentkezéskor, vagy első jelszó változtatáskor használt színt használtam fel a weboldalhoz. Így elérve azt, hogy egységes legyen.

6. Összegzés

Összegészében a kijelölt terveket teljesítettem magammal szemben, amit a terveknél kigondoltam. Egyes funkciókat frissíteni szeretném és ahogy a fejlesztési tervekben is írtam, azokat viszont meg szeretném megvalósítani a közel jövőben. A weboldal és az asztali közötti kapcsolatot jól megoldottam véleményem szerint. Így még se lehet egymástól elválasztani, hiszen egyik függ a másiktól.
A weboldalon a gyerek követés elég egyedinek számít egy nevelő intézetben. Sajnos nem hallottam még, hogy alkalmazzák bár hol is. De egy új lehetőséget tudna nyitni, akár plusz funkciók hozzáadásával.
Ellátás is fontos, hogy a szülő tudjon a gyermeke lelki és fizikai bajáról is. Legalább is az igény meg van a szülők felől.
A grafikai megjelenéssel nagyon elégedett vagyok, eleinte nem gondoltam, hogy ennyire jól fog menni. Egy kis idő bele adásával minden lehetséges.
A programozást objektum orientálton csináltam. Próbáltam a tiszta kód elvét is használ a kódok megírásával kapcsolatában.
Össze egészében elégedett vagyok a munkával. Sok mindent adott nekem, mind fejlődésben mind egy munka mellett való kitartásban.

7. Köszönet nyilvánítás

Szeretném megköszöni Kádár Tünde tanárnőnek az adatbázis tanulmányokat, Gyuris Csaba tanár úrnak az C# programozási nyelvben a tanításokat és valamint Bálint Róbert tanár úrnak a webfejlesztésbe való sok-sok támogatását.

8. Plágium nyilatkozat

Alulírott, Bálint István aki az SZSZC Vasvári Pál Gazdasági és Informatikai Szakgimnázium tanulója (szoftverfejlesztő szakon), hogy az általam készített Live In Care asztali alkalmazás és a web oldal hozzá a saját munkám része.

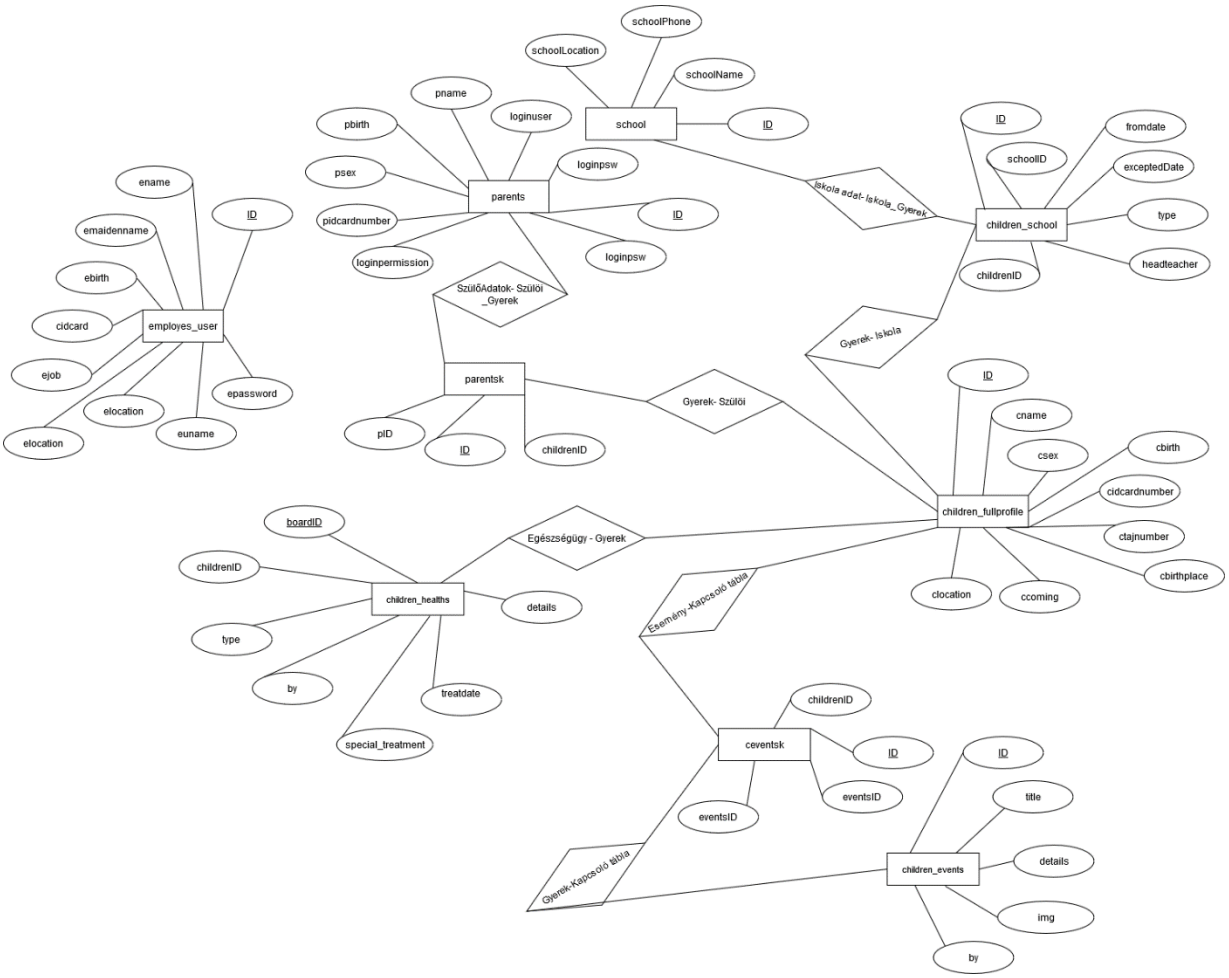
Szeged, 2020.04.08.

…….

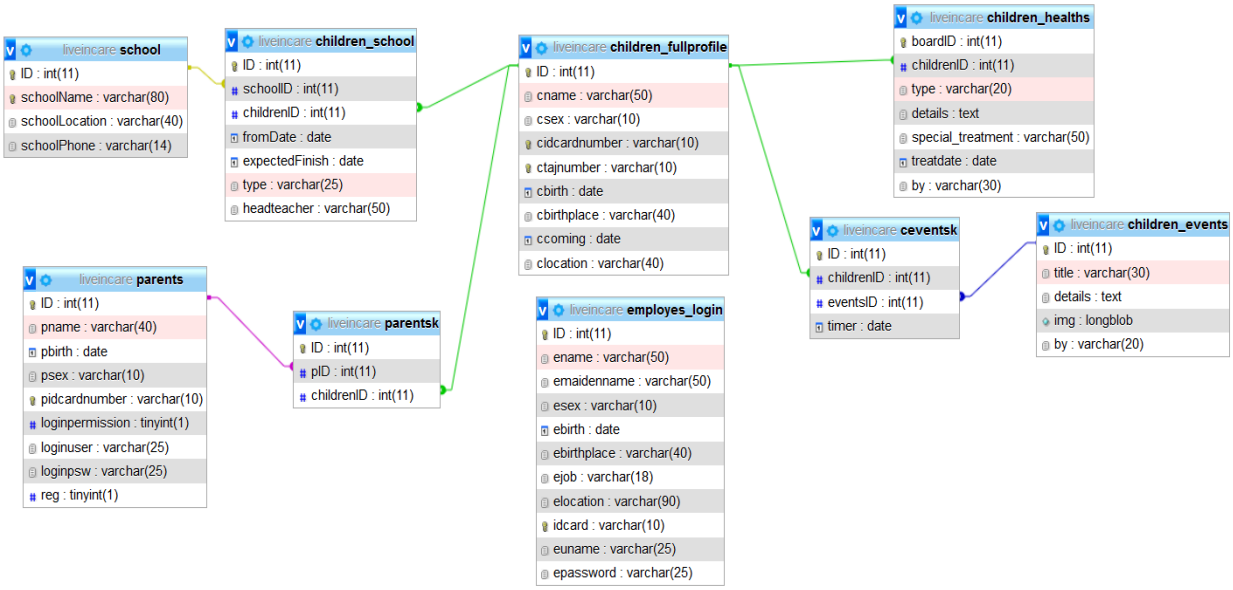
Bálint István

9. Mellékletek:

I. E-K diagramm (I. ábra)



II. Bachmann-ábra (II. ábra)



III. Use-Case diagramm (III. ábra)