**Debreceni Szakképzési Centrum Beregszászi Pál Technikum**

MikorDolgozom munkahelyi beosztás-kezelő weboldal fejlesztése



**Készítette:**

Tisza-Kis Eleonóra

**Csapattagok:**

Ferenczik Judit, Szmolka László

**Témavezető:**

Boros Sándor

Debrecen, 2022

**Tartalomjegyzék**

[1 Bevezető 4](#_Toc99980284)

[2 Megoldandó feladatok 5](#_Toc99980285)

[3 Fejlesztői dokumentáció 7](#_Toc99980286)

[3.1 Fejlesztői eszközök 7](#_Toc99980287)

[3.2 Felületterv 8](#_Toc99980288)

[3.3 Adatmodell leírása 9](#_Toc99980289)

[4 Részletes feladatspecifikáció 14](#_Toc99980290)

[4.1 A program lényeges függvényeinek, eljárásainak specifikációja 14](#_Toc99980291)

[4.1.1 A program legfontosabb változói 14](#_Toc99980292)

[4.1.2 A weboldalhoz szükséges forrásfájlok szerkezete 15](#_Toc99980293)

[4.2 Algoritmusok az oldalon 18](#_Toc99980294)

[4.2.1 Bejelentkezés 18](#_Toc99980295)

[4.2.2 A naptár megjelenítése 18](#_Toc99980296)

[4.2.3 A kérelmek elküldése 19](#_Toc99980297)

[4.2.4 A szabadság igények elfogadása 19](#_Toc99980298)

[5 Tesztelési dokumentáció 20](#_Toc99980299)

[5.1 Felhasználói funkciók tesztelése 20](#_Toc99980300)

[5.2 Egységtesztek 22](#_Toc99980301)

[6 Továbbfejlesztési lehetőségek 23](#_Toc99980302)

[7 Felhasználói dokumentáció 24](#_Toc99980303)

[7.1 Hardver és szoftver követelmények 25](#_Toc99980304)

[7.2 A program telepítésének, konfigurálásának leírása 25](#_Toc99980305)

[7.3 A program használatának részletes leírása 25](#_Toc99980306)

[7.3.1 Bejelentkezés 25](#_Toc99980307)

[7.3.2 Naptár 26](#_Toc99980308)

[7.3.3 Kérések oldal 26](#_Toc99980309)

[7.3.4 Értesítések 27](#_Toc99980310)

[7.3.5 Beosztottjaim oldal 27](#_Toc99980311)

[7.3.6 Személyes 28](#_Toc99980312)

[8 Összegzés 29](#_Toc99980313)

[9 Irodalomjegyzék, forrásjegyzék 30](#_Toc99980314)

# Bevezető

A szoftverfejlesztő- és tesztelő szak kimeneti követelményeként előírt projektmunkában hárman vettünk részt: Ferenczik Judit, Tisza-Kis Eleonóra és Szmolka László Tibor. László könyvelőként dolgozik, és saját szakmai tapasztalatai alapján vetette fel az ötletet, hogy egy munkahelyi beosztáskezelő programot hozzunk létre. A szoftverpiacon jelenleg igénybe vehető szolgáltatások, alkalmazások átláthatatlanok és nehezen kezelhetők egy átlag felhasználó számára. Ezért egy könnyen kezelhető, letisztult, felhasználó-barát szoftver készítéséhez kezdtünk el ötleteket gyűjteni. A program alkalmas lenne arra, hogy egy cég vagy csoport vezetői egy helyen hozzáférhessenek a dolgozóik adataihoz, meg tudják adni előre a munkabeosztásukat, szabadnapjaikat. Ugyanakkor a munkavállalók is hozzáférést kapnak a szoftverekhez, így ők is láthatják beosztásukat, melyek a munkanapok és melyek a pihenő- vagy ünnepnapok. A felületen kérhetik beosztásuk módosítását, illetve szabadnap igényeiket is leadhatják egyúttal. Az esetleges táppénzt is itt tudják jelölni, így az adott hónapot követően átláthatóvá válik, hogy hány munkanapot dolgozott az adott személy, ami a bérszámfejtők segítségére lehet.

A csapatban mind egyetértettünk, hogy egy olyan szoftvert hozzunk létre, ami a való életben is hasznos, és ami a későbbiekben akár a piacon is megállná a helyét. Leegyszerűsítheti a munkavégzés előkészítését, könnyű elérést biztosítva a dolgozóknak és főnökeiknek a munkaszervezéshez, akár otthonról, akár útközben. Ezért döntöttünk úgy, hogy belekezdünk ebbe a projektbe.

A célunk, hogy háromféle program összekapcsolásával minél könnyebben és gyorsabban tudják a beosztásokat kezelni mind a munkáltatók és a munkavállalók. A feladatot három részre osztva oldjuk meg, miközben összekötjük egy közös adatbázissal, így egymást segítve, ugyanakkor önállóan hozunk létre egymástól független programokat.

Dolgozatomban a vizsgaremek elkészítésének részleteit szeretném bemutatni. Az általunk választott feladat egy munkahelyi beosztás-kezelő program három platformra történő fejlesztése, ehhez kapcsolódóan fejtem ki, hogyan hoztunk létre egy weboldalt és egy Android alkalmazást erre a célra.

# Megoldandó feladatok

A projektmunka első fázisaként arról egyeztettünk, hogy pontosan milyen programot szeretnénk, és milyen funkciók legyenek benne elérhetők. Végeredményként egy letisztult, egyszerű felület megvalósítását terveztük, tudásunk, képességeink legjavát felhasználva, amely stabilan működő alkalmazásokat eredményez.

Ezt követően elosztottuk a feladatokat egymás között. Mivel hárman dolgozunk a projekten, adott volt, hogy három felületre egy-egy személy készítse el az alkalmazást. Judit az Android-alkalmazást vállalta, László az asztali alkalmazást, én pedig a webes felület elkészítésére jelentkeztem. Sajnos azonban az asztali alkalmazás végül nem került fejlesztésre.

A következő fázisban a felhasználók körét határoztuk meg. Végleges megoldásként kétféle felhasználói jelleget különböztettünk meg: alkalmazotti és vezetői felület. Lehetséges opcióként felmerült egy harmadik, „admin” felület létrehozása is, aki felügyeli a felhasználók tevékenységét, de végül ezt elvetettük, mivel csupán az adatbázis kezeléséhez lett volna szükséges.

Ezek után a felhasználók jogosultságait, az általuk elérhető funkciókat definiáltuk. Az alkalmazott részéről a legfontosabb a munkabeosztásának ellenőrzése, esetlegesen annak változtatásának lehetősége. Ehhez kapcsolódóan szükséges a „Kérések” menüpont, ahol a szabadság és táppénzt jelölhető/kérelmezhető, valamint egy „Értesítések” menüpont, ahol a jóváhagyott vagy elutasított kérelmei láthatók. Mivel az alkalmazás egyik célja a jelenlétek rögzítése, ezért a jelenlétek megtekintésére is szeretnénk egy felületet. Végül pedig a személyes adatainak módosítására is lehetőséget szerettünk volna biztosítani.

Az alkalmazotti felülethez hasonló a beosztást kezelő vezetői felület. Megtekintheti az alá tartozó beosztottak listáját és beosztásukat. A „Kérések” menüpontban a hozzá beérkező szabadság kérelmeket láthatja, és egyúttal jóvá is hagyhatja. A személyes adatok módosítására a vezetői jogosultsággal is lehetőség van.

Mindezen feladatok megoldásához a következő lépés a közös adatbázis megtervezése volt. Hosszas egyeztetés után jött létre a végleges adatszerkezet, amely alkalmas az összes személy (akármilyen jogosultsággal rendelkezik is) adatainak tárolására, és az összes funkció működtetéséhez: egy naptár, amely a munkanapokat és munkaszüneti napokat elkülöníti, egy tábla a kérések kezeléséhez.

A program működéséhez tehát fontos egy bejelentkező felület, ahol e-mail cím és a megfelelő jelszó használatával lehet az oldal tartalmát megtekinteni. Egyúttal itt dől el, hogy alkalmazotti vagy vezetői felületre kerül a felhasználó. Jogosultságtól függetlenül szükséges, hogy az adataiban történő változást azonnal végrehajtsa az adatbázisban is, például az alapértelmezett jelszót a saját kódjára lecserélje a felület használója.

A beosztások megtekintéséhez egy naptár funkció bevezetésére van szükség: az adatbázissal való kapcsolat segítségével a hétvégék és munkaszüneti napok jól elkülönülnek, valamint felhívja a figyelmet a bedolgozandó pihenőnapokra, azaz a szombati munkavégzésre. A munkaszüneti és hétvégi napokra szabadság kérése sem lehetséges, így az algoritmus működésének is ehhez kellene igazodnia.

A szabadság-kérelmek, valamint a táppénz-bejelentés információit szintén az adatbázis megfelelő táblájába kell rögzíteni, hogy az majd az értesítésekben is megjelenhessen. Az eltárolt adatok alapján kap jelzést az adott munkavállaló vezetője, és a dolgozó is az elfogadás tényéről. Ezt követően lehet eltárolni az adatokat a munkavégzésről, hogy a megtekinteni kívánt hónapban mely napokon dolgozott, és melyek azok a napok, amelyeken nem történt munkavégzés.

Ezeknek a feladatoknak a megvalósítása a közös cél, ami alkalmazásonként más programozási megoldásokat kíván. Hasonló, egymással kompatibilis programok létrehozása a cél mobilos, webes és asztali felületre.

# Fejlesztői dokumentáció

A csapatunk által vállalt projektfeladatban a webalkalmazás létrehozásának, felépítésének részleteit szeretném bemutatni. Az eddig tanult, webfejlesztésben használt eszközök, valamint adatbázis-kezelési ismeretek segítségével igyekeztem létrehozni a beosztás-kezelő weboldalt, a MikorDolgozom felületét.

## Fejlesztői eszközök

A weboldal fejlesztése az Apache Netbeans IDE 12.5-ös integrált fejlesztői környezetben történt. Ez a nyílt forráskódú program több programozási nyelvet (Java, PHP, JavaFX, JavaScript) és keretrendszert is támogat. Számomra ez a legkönnyebben használható program, beépített eszközei nagyban megkönnyítik és gyorsítják a kódok írását. A gépelés során felajánlott lehetőségek, a táblázatok, listák gyors beszúrási lehetősége, a különböző sablonok mind olyan funkciók, amik hasznosak a fejlesztés során. A webprogramozás során php és html kódokat használtam, a design kialakításához pedig lépcsőzetes stíluslapokat (CSS) és a Bootstrap keretrendszert alkalmaztam.

Az adatszerkezetet a phpmyadmin felületén hoztuk létre, amelynek futtatásához a XAMPP Control Panel v3.3.0-ás verzióját használtam. Itt kerültek kialakításra a táblák és kapcsolataik, valamint az adatok egy részének feltöltése – bizonyos adatok pedig a weboldal működtetése során kerülnek az adatbázisba.

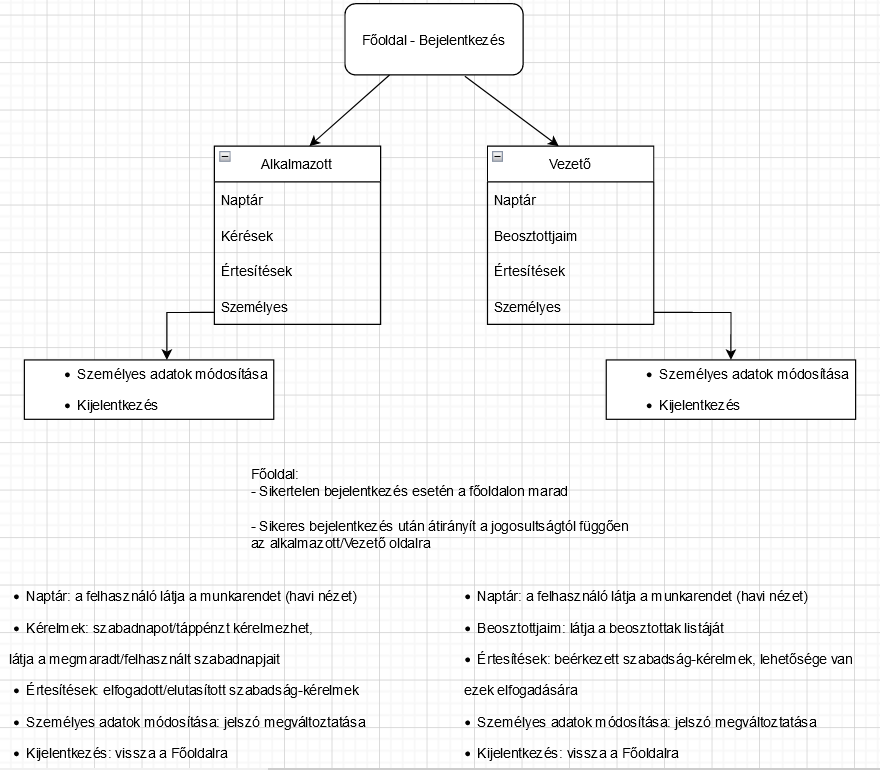
Az adatbázis megtervezéséhez és a weboldal funkcióinak ábrázolásához a diagrams.net/draw.io webes felületet használtuk. Az oldal lehetővé teszi folyamatábrák elkészítését, adatkapcsolatok jelölését. Közvetlen kapcsolat létesíthető egy Github fiókkal is, így könnyen és azonnal meg lehet osztani a projektben résztvevőkkel a diagramokat.

A közös munka során a GitHub szoftverfejlesztési verziókövetés-szolgáltatását vettük igénybe. Így a projekt összes résztvevője hozzáférést kap a kódokhoz, fájlokhoz, segíti az együttműködést, az ötletek, elkészült programrészek azonnali megosztását. Ezen kívül biztonságot is nyújt az elkészült anyagok elvesztése ellen. A megosztáshoz a GitHub asztali applikációját használom, amely azonnal jelzi a fájlokban történt változást és felkínálja a feltöltés lehetőségét.

A vizsgaremek dokumentációjához a Microsoft Office 365 alkalmazáscsomagból a Word szövegszerkesztőt használom, hogy a követelményeknek megfelelő szerkezetű, megformázott szöveg jöjjön létre. A Word segítséget nyújt, hogy az elvárásoknak eleget téve helyesen legyen tagolva a szöveg, a címsorok alkalmazásával áttekinthető legyen, megkönnyítve a tartalomjegyzék generálását is. Lehetővé teszi a szöveg és a kódrészletek elkülönítését, képek, ábrák beillesztését, valamint a helyesírás ellenőrzésében is segítséget nyújt.

A vizsgaremek elkészítése során használt összes program és keretrendszer legális, többsége nyílt forráskódú, valamint az intézmény által a tanulóknak ingyenesen biztosított Microsoft Office 365 is jogtiszta.

## Felületterv

A felület megtervezésekor az elsődleges szempont a felhasználói funkciók elkülönítése volt. A kétféle jogosultság alapján (alkalmazott, vezető) a weboldal megjelenése is kettéválik, hogy a felhasználó csak a jogosultságának megfelelő funkciókat érhessen el.

. ábra: Diagram a felület felépítéséről

Az elsődlegesen megjelenő oldalon van lehetőség a bejelentkezésre: itt e-mail cím és a hozzá tartozó jelszó segítségével lehet belépni, ez alapján dől el, hogy melyik felhasználói oldalra irányít át a rendszer.

Alkalmazottként megtekintheti az alapértelmezett munkarendet, amely egy hónapokra lebontott naptár. Ez tartalmazza az aktuális év munkanapjait, pihenő-és ünnepnapjait, valamint a munkanap áthelyezéseket. Ez vonatkozik minden munkavállalóra.

Minden ettől eltérő igényt a *„Kérelmek”* oldalon tud rögzíteni. Két dátumválasztó segítségével lehet egy időintervallumot megadni, amikor szabadságot vagy táppénzt/betegszabadságot szeretne jelezni. Az eltávozás okát szintén itt tudja bejelölni a fent említett lehetőségek közül választva.

A munkavállaló által igényelt szabadnapok elbírálásáról az *„Értesítések”* fül alatt lehet tájékozódni. Minden szabadság-igényről megtekinthető, hogy a főnöke elfogadta-e a kérelmét. A táppénz nem kerül elbírálás alá, így az nem jelenik meg ezen a felületen.

A *„Személyes”* menüpontban a felhasználói fiókkal kapcsolatos tevékenységekre van lehetőség. Itt tudja megváltoztatni a jelszavát, valamint kijelentkezni a weboldalról.

Főnökként történő bejelentkezés esetén hasonló funkciók vannak, mint alkalmazottként. Szintén megtekintheti a *„Naptár”* felületet ugyanolyan formában, mint a beosztottak.

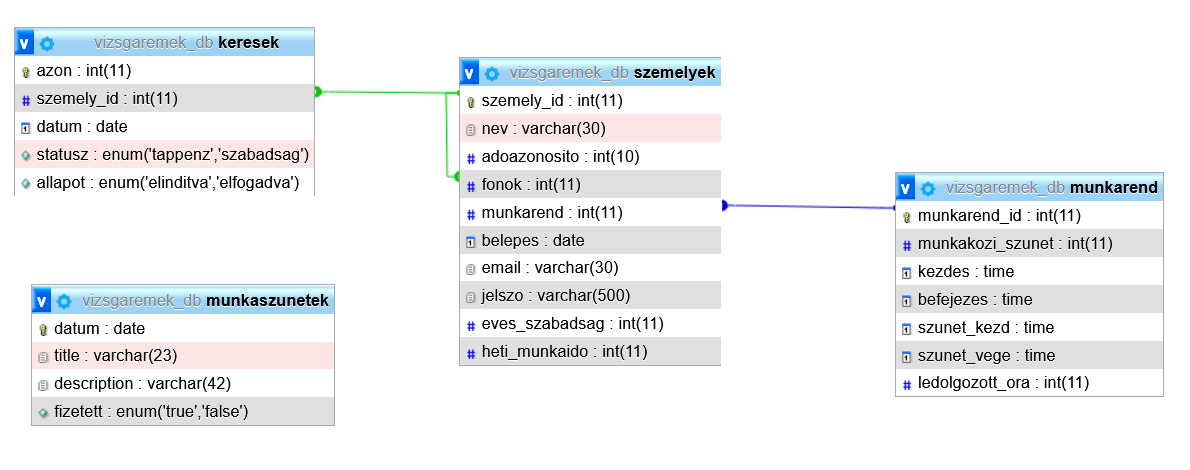
A *„Beosztottjaim”* fülön tekinthető meg kilistázva, hogy melyek azok a dolgozók, akik az ő részlegéhez tartoznak, akiknek a közvetlen felettese.

A dolgozói által kérelmezett szabadságokat az *„Értesítések”* oldalon tekintheti meg: melyik beosztottja mely napokra szeretne szabadnapot kivenni. Egyúttal ezen a felületen el is tudja ezeket fogadni, amit az alkalmazottak szintén ezen az oldalon tekinthetnek meg.

A *„Személyes”* oldalra belépve ugyanazok a funkciók érhetők el, mint alkalmazottként.

## Adatmodell leírása

A weboldal működtetéséhez elengedhetetlen egy adatbázis használata, az oldalak megjelenítéséhez, az adatok kiírásához először el kell tárolni minden szükséges információt. Szükség van olyan táblákra, amelyek előre meghatározott adatokat tartalmaznak, valamint a weboldal funkcióinak használatából kinyert adatok tárolására is létre kellett hozni táblákat. Összesen 4 táblából álló adatszerkezetre van szükség az oldal működéséhez.



. ábra: Adatbázis kapcsolatok

Az adatokkal előre feltöltött táblák a *„szemelyek”*, *„munkarend”* és a *„munkaszunetek”*.

A képen asztal látható

Automatikusan generált leírás

. ábra: A "szemelyek" tábla szerkezete

A *„szemelyek”* tábla tartalmazza az összes dolgozó személyes adatát, ami alapján meg lehet különböztetni az egyes felhasználókat. A *„szemely\_id”* mező egy egyedi érték, numerikus adattípus (*int*) egyúttal az elsődleges kulcs is, és az *auto\_increment* funkció segítségével biztosítható, hogy minden felhasználónak más legyen az azonosítója. A *„nev”* mező tárolja a felhasználó teljes nevét, amely *varchar* típusú. Szintén a megkülönböztetést segítendő, található egy *int* típusú *„adoazonosito”* érték, hogy a nevek esetleges egyezése miatt elkülönüljenek a felhasználók. Fontos adatot tartalmaz a *„fonok”* mező, ami idegen kulcs megszorításként kapcsolatban áll a *„szemely\_id”-*val. Ez tartalmazza az adott dolgozó főnökének azonosítóját, így képesek vagyunk tárolni, ki melyik részlegen dolgozik. Az a dolgozó, aki vezető beosztásban van, nem kap ilyen azonosítót, a *„fonok”* mező ilyenkor üresen marad. A megszorítás pedig lehetővé teszi, hogy csak létező *„szemely\_id”* kerüljön rögzítésre főnökként. A *„munkarend”* egy *int* típusú adat, a *„munkarend”* táblával kap értelmet. Értéke 0 vagy 1, attól függően, hogy az adott dolgozó kötött vagy kötetlen munkarendben dolgozik. A *„belepes”* egy *date* típusú érték, a dolgozó munkába állásának dátuma. Az *„email” varchar* típusú adat, a dolgozó által munkába állaskor megadott e-mail cím, az azonosítás alapja, ezzel tud belépni a felhasználói felületére. Ehhez kapcsolódik a *„jelszo”* mező is, ami a felhasználói fiók létrehozásakor egy alapértelmezett érték, ezt később a felhasználó tetszés szerint bármikor megváltoztathatja. A biztonságos adatkezelés érdekében a jelszavak SHA1-es titkosítással kerülnek rögzítésre.[[1]](#footnote-1) Az *„eves\_szabadsag”* mezőben az évente felhasználható szabadnapok száma (*int*) látható, ez szükséges ahhoz, hogy tudjuk, még igényelhet-e a felhasználó szabadságot. A *„heti\_munkaido”* szintén numerikus, *int* típusú, a hetente ledolgozandó órákat tartalmazza.



Automatikusan generált leírás

. ábra: A "munkarend" tábla szerkezete

A *„munkarend”* tábla a kétféle munkarend jellemzőit tárolja. Ezek megkülönböztetésére szolgál az *int* típusú *„munkarend\_id”,* amely ebben az esetben 0 és 1 lehet. Egységes jellemzői az egy napra kiadandó szünet hossza, valamint a naponta ledolgozandó órák száma *(„munkakozi\_szunet”*, *„ledolgozott\_ora”)*, ezek minden munkavállalóra vonatkoznak*.* Mindkettő numerikus értékű, *int* típusú. Ezeken kívül a munkaidő és a munkaközi szünet kezdetének és végének időpontját szükséges rögzíteni, ezek *time* típusú adatok (*„kezdes”, „befejezes”*, *„szunet\_kezd”*, *„szunet\_vege”*). Az alkalmazottként nyilvántartott, azaz kötött munkaidővel rendelkező felhasználók számára ezek rögzített értékek, egységesen dolgoznak, míg a vezető beosztásúak kötetlen munkaidővel rendelkeznek.

A képen asztal látható

Automatikusan generált leírás

. ábra: A "munkaszunetek" tábla szerkezete

A *„munkaszunetek”* tábla szintén előre feltöltésre került az aktuális évre vonatkozó adatokkal. Ez tartalmaz minden egyes kivételt, amikor a megszokottól eltérően van (vagy nincs) munkavégzés, azaz az összes ünnepnap, a hosszú hétvégék miatti pihenőnapok, és a bedolgozandó munkanapok. Ezeknek az adatoknak a tárolásához szükség van egy *date* típusú *„datum”* mezőre, ami a kivételt képező dátumot tárolja. A *„title” varchar* típusként rögzíti az ünnep nevét, vagy ha áthelyezett munkanap esik az adott dátumra. A *„description”* mező szintén karaktersorozatként jelenik meg, az adott nap részletesebb jellemzőit adja meg (pl. „áthelyezett munkanap március 14. helyett”). A *„fizetett”* oszlopban *enum* típusban *true* és *false* lehetőségek közül választva lehet megadni, hogy fizetett-e az adott ünnepnap.

A képen asztal látható

Automatikusan generált leírás

. ábra: A "keresek" tábla szerkezete

A *„keresek”* tábla az oldal egyik fő funkciójának működését biztosítja: lehetővé teszi a benyújtott szabadság-kérelmek rögzítését, illetve jelezheti a dolgozó, mikor van betegszabadságon, táppénzen. Minden egyes napra benyújtott kérelem egyedi azonosítóval (*„azon”)* rendelkezik, szintén az *auto\_increment* funkció segítségével kap minden adat egy új *int* értéket. A *„szemely\_id”* a *„szemelyek”* táblával biztosítja a kapcsolatot, így kerül dokumentálásra, hogy melyik felhasználó adott be kérelmet. A *„datum”* mezőben található *date* érték tárolja, hogy mely napokat érinti a kérelem. A távollét okát a *„statusz”* mező mutatja meg, *enum* formátumban a *’tappenz’* és a *’szabadsag’* lehetősége választható. Az *„allapot”* szintén *enum* formátumban tárolja, hogy elfogadásra került-e a kérelem, ennek értéke *’elinditva’* (tehát beérkezett a kérelem) vagy *’elfogadva’* (a vezető jóváhagyta) lehet.

# Részletes feladatspecifikáció

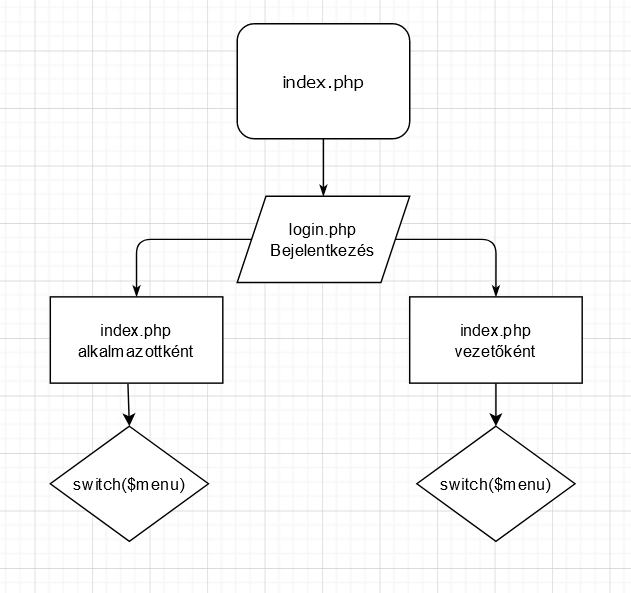
## A program lényeges függvényeinek, eljárásainak specifikációja

### A program legfontosabb változói

A weboldal megfelelő működéséhez több változó bevezetésére is szükség van.

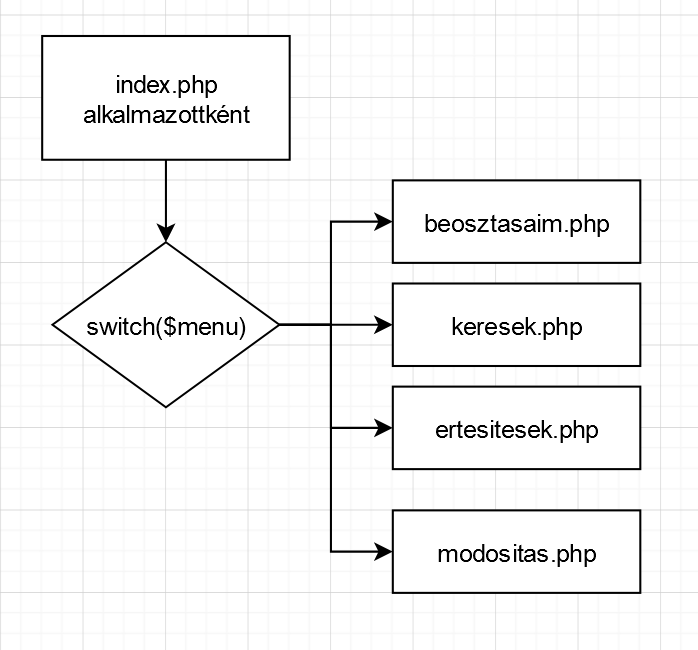
* $\_SESSION[„login”]: a weboldal indításakor először azt szükséges vizsgálni, hogy be van-e jelentkezve egy felhasználó. Ez igaz vagy hamis értéket vehet fel, alapértelmezetten hamis értékkel indul az oldal.
* $\_SESSION[„jog”]: a belépést követő jogosultságot ellenőrzi, azaz hogy milyen minőségben van bejelentkezve a felhasználó, ettől függ, hogy milyen felületet lát, aki belép. Kétféle megjelölést kaphat, *„fonok”* vagy *„alkalmazott”*, ennek függvényében változik, hogy milyen szolgáltatásokat érhet el a dolgozó.
* $email és $jelszo: ez a két változó szükséges a bejelentkezéshez, megadásuk kötelező, ha bármelyik hibásan van megadva, hibaüzenet jelzi. A weboldal működése szempontjából a továbbiakban nincs rá szükség, a $\_SESSION[„login”] változó tárolja, hogy sikeresen belépett az adott személy.
* $\_SESSION[„szemely\_id”]: a belépést követően nem csupán a bejelentkezés tényét kell rögzíteni, hanem fontos, hogy melyik személy kívánja használni az oldalt. Ez nem csak a jogosultság miatt szükséges, hanem hogy a személyre szóló adatokat lássa a felhasználó. Főnökként csak a saját dolgozóira vonatkozó információkat, dolgozóként pedig csak az őt érintő kérdéseket.
* $\_POST[] szuperglobális változók: ezek a változók az űrlapok kitöltésével keletkező adatok begyűjtéséhez szükségesek, amiket később továbbítani lehet a feldolgozó programokhoz, majd az adatbázisba.

### A weboldalhoz szükséges forrásfájlok szerkezete

Az oldal működtetéséhez 18 fájl került létrehozásra, ezek többnyire php fájlok, valamint a formázáshoz szükséges stíluslapok. Létrejött egy fájl a használt függvények tárolására, ez elsősorban a tesztelés miatt volt indokolt. Nem minden fájlra van szükség egy adott felhasználó esetében, a különböző felhasználói minőségek miatt. Némelyik php-oldal az adatok megjelenítéséért felel, de vannak olyan oldalak, amelyek csak a háttérben futnak, segítve az adatok beolvasását vagy kiírását.

7. ábra: Flowchart I.

Az *index.php* elindulását követően, mivel a $\_SESSION[„login”] változó értéke hamis, a *login.php* oldalra kerül a felhasználó, ahol e-mail cím és jelszó megadásával léphet be. A megfelelő adatok elküldésével sikeresen bejelentkezik a felhasználó, és visszakerül az *index.php* oldalra. Itt a felső menüsor segítségével tud navigálni a különböző oldalak között. Már *az index.php*-ban meghívásra kerül a *head.php* fájl, ami egy html kód „head” részletét tartalmazza. Ez magába foglalja a stíluslap linkjét, az *indexcss.css*-t is. A *login.php* oldal hívja meg az *ellenorzes.php* fájlt, ami a bejelentkező adatok ellenőrzését végzi el. Ehhez előbb csatlakoznia kell a vizsgaremek adatbázisához, amit a *connect.php* végez el.

Alkalmazotti bejelentkezés:

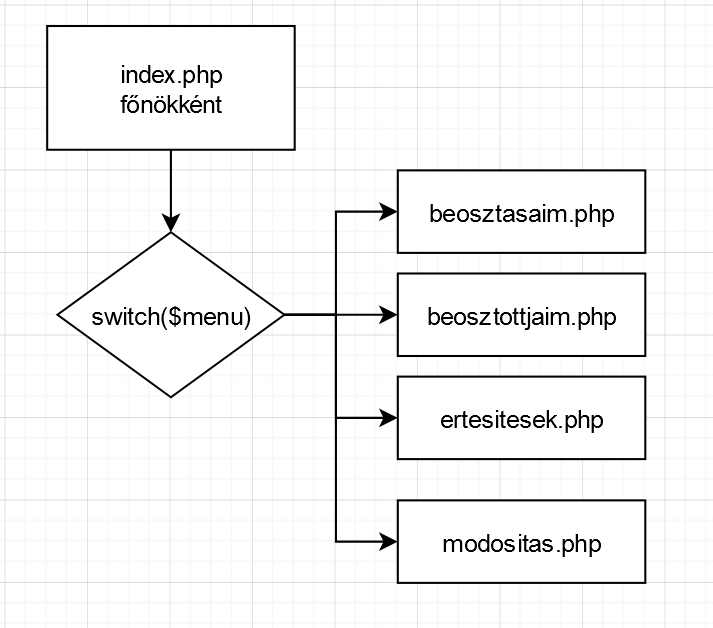
8. ábra: Flowchart II. - alkalmazott

A NAPTÁR menüpontot választva a *beosztasaim.php* oldalra kerül a felhasználó. Ez hívja meg a *naptar.php* fájlt, valamint a megjelenítéshez kapcsolódó stíluslapot, a *naptar.css*-t. Ezek segítségével jelenik meg egy naptár az aktuális évre vonatkozóan, hónapokra lebontva. A *naptardb.php* fájlból pedig meghívja a függvényt, amely az adatbázisban eltárolt munkaszüneti napokat és az áthelyezett munkanapokat írja ki.

A KÉRÉSEK menüpontra kattintva egy űrlap jelenik meg, ehhez kapcsolódik a *kerelem\_kuld.php* oldal. Ez ismét tartalmazza *a connect.php* fájlt, hogy az adatbázishoz való csatlakozás által az űrlapon küldött adatok mentésre kerüljenek.

Az ÉRTESÍTÉSEK fület választva az alkalmazottként belépő felhasználó az adatbázisból kiolvasott adatok alapján tudja megtekinteni, hogy milyen választ kapott a kéréseire. Ehhez szintén szükség van *a connect.php* fájlra, hogy kapcsolatot teremtsen az adatbázis megfelelő táblájával.

A SZEMÉLYES menüpont felé irányított egérrel jelennek meg az ADATOK MÓDOSÍTÁSA, valamint a KIJELENTKEZÉS gombok. Előbbire klikkelve a *modositas.php* oldalra irányítja a felhasználót, ahol egy űrlap várja. Az űrlapon kitöltött adatok továbbításához szükséges a *jelszo\_mod.php* fájl, amely szintén a *connect.php* segítségével kapcsolódik az adatbázishoz. Utóbbit választva pedig törölhető az aktuális munkamenethez tartozó összes adat, így fiókjából kijelentkeztetve visszairányít az *index.php-ra*.

Bejelentkezés vezetőként:

9. ábra: Flowchart III. - vezető

A NAPTÁR felületre kattintva főnökként is a *beosztasaim.php* oldalra kerül a felhasználó, ugyanazokkal a funkciókkal, mint alkalmazottként.

A BEOSZTOTTJAIM menüpontot kiválasztva a *beosztottjaim.php* oldalra lehet eljutni. Ezen a felületen tudja megtekinteni a részlegéhez tartozó alkalmazottak listáját, melyet a *connect.php* segítségével olvas ki a program az adatbázisból.

Az ÉRTESÍTÉSEK fülre klikkelve szintén az *ertesitesek.php* oldal nyílik meg, ahol főnökként a beosztottjai által elküldött kérelmei láthatók. Ehhez szükséges ismét kapcsolatot létesíteni az adatbázissal a *connect.php* által, így kerülnek kiírásra a kérések egy táblázatban. A Küldés gombra kattintva újra kapcsolatot kell teremteni az adatbázissal a *szab\_elfogad.php* oldal által, amely segítségével rögzíti az adott táblában az elfogadás tényét.

A SZEMÉLYES menüpontban pedig ugyanazok a funkciók érhetők el, mint alkalmazottként.

## Algoritmusok az oldalon

Ebben a fejezetben a weboldalon működő legfontosabb algoritmusokat, függvényeket szeretném bemutatni.

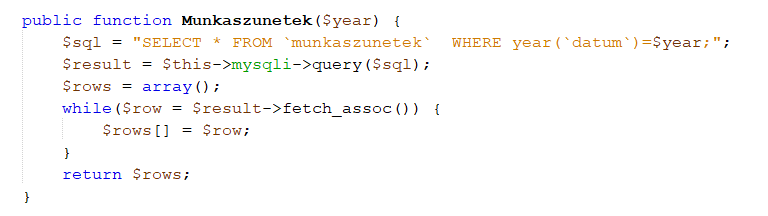
### Bejelentkezés

A *login.php* oldalon megadott adatok alapján indul el a bejelentkezés folyamata, ami az *ellenorzes.php* által történik meg. Ennek főbb lépései:

* annak ellenőrzése, hogy a Belépés gombra kattintva kerültek-e be adatok. Ha bármelyik mező üresen marad, visszairányít a *login.php* oldalra és hibaüzenetet küld.
* SQL-utasítás előkészítése: az utasítás, amely megkeresi, hogy megtalálhatók-e a megadott adatok az adatbázisban.
* e-mail cím ellenőrzése: az adatbázisban létezik-e a megadott e-mail cím. Ha igen, eltárolja a hozzá tartozó adatokat, ha nem, akkor hibaüzenetet jelez.
* jelszó ellenőrzése: sha1() függvénnyel átalakítja a jelszót, és ellenőrzi, hogy egyezik-e az e-mail címhez tartozó jelszóval. Ha egyezést talál, $\_SESSION[] változókban eltárolja a felhasználó adatait, mint a *„szemely\_id”,* név. A $\_SESSION[„login”] ekkor igaz értéket vesz fel, a $\_SESSION[„jog”] értékét pedig a felhasználó minőségének megkeresésével „alkalmazott” vagy „fonok” értékre állítja.

### A naptár megjelenítése

A naptár kiíratása a *beosztasaim.php* oldalon történik, amihez szükség van a naptar.php és a *naptardb.php* fájlokra. Ehhez kell meghívni a *Munkaszunetek($year)* függvényt, ami az adatbázisból kiválasztja a megfelelő napokat.



. ábra: Munkaszunetek() függvény

A *function havi($year, $month)* segítségével írjuk ki a hónapok neveit, majd ezután a hét napjainak megfelelően a dátumokat is.

A képen szöveg látható

Automatikusan generált leírás

. ábra: Naptár kiíratása - kódrészlet

### A kérelmek elküldése

A szabadság igénylésének és a betegség miatti távollét rögzítésének folyamata a *kerelem\_kuld.php* oldalon keresztül:

* annak ellenőrzése, hogy lettek-e adatok küldve a Küldés gomb által
* a kezdő dátum mentése a *$start* változóba
* a távollét okának megállapítása – szabadság (ennek állapota „elinditva” lesz) vagy táppénz (automatikusan elfogadott)
* annak ellenőrzése, hogy van-e záró dátum (*$end* változó). Ha igen, akkor a *$start* és *$end* dátumok közé eső periódusból minden napra beilleszt a *„keresek”* táblába egy sort a megfelelő dátummal, a távollét okával és annak állapotával az igénylő személy id-jához.

### A szabadság igények elfogadása

A szabadságok engedélyezése a *szab\_elfogad.php* oldalon történik:

* az *ertesitesek.php* oldalon bejelölt checkboxok azonosítóinak tárolására létrejön az *$azonositok* tömb
* ha a Küldés gombra kattintás előtt nem választott ki egy jelölőnégyzetet sem, akkor a tömb üresen marad, amire hibaüzenetet küld
* egy foreach ciklussal minden bejelölt azonosítót hozzáad a tömbhöz
* a tömb tartalmának hozzáadása egy SQL-utasításhoz, amely felülírja a szabadságok állapotát ’elfogadva’ értékre.

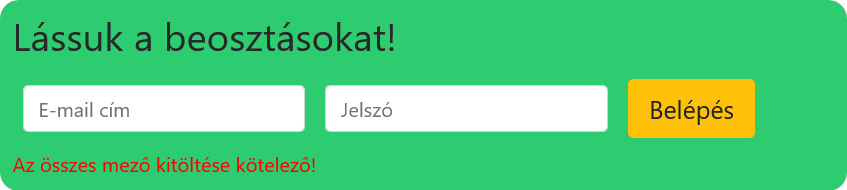
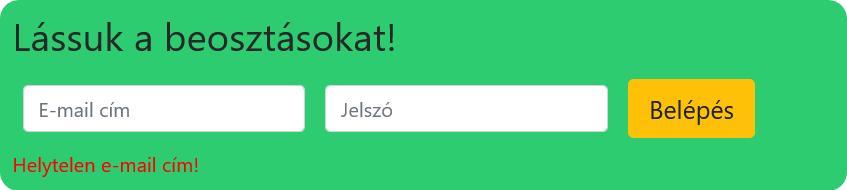
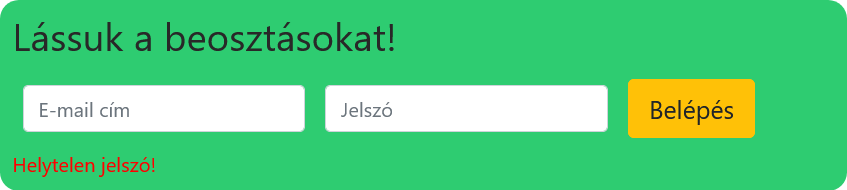
# Tesztelési dokumentáció

A tesztelés során egyrészt arról szeretnénk meggyőződni, hogy a weboldal teljesíti-e az elvárt funkciókat, helyesen futnak-e az algoritmusok, másrészt azt ellenőrizzük, hogy megfelelően kezeli-e a hibás bemeneteket, védve van-e a rosszindulatú használat ellen, eléggé „bolondbiztos”-e.

## Felhasználói funkciók tesztelése

Ezek ellenőrzéséhez végigjártam az oldalakat és funkciókat, kipróbálva mindent egy felhasználó szemszögéből.

Bejelentkezés:

* az üresen hagyott űrlapot nem küldi el, hibaüzenetet jelez, mely szerint minden mezőt kötelező kitölteni
* ha az adatbázisban nem létező e-mail címet ad meg a felhasználó, a rendszer üzenetet küld „Helytelen e-mail cím!” felirattal
* ha az e-mail címet megfelelően adta meg, de a jelszó nem egyezik az adatbázisban tárolttal, szintén hibát jelez.

12. ábra: Hibaüzenetek bejelentkezéskor

Kérelem:

* ha a felhasználó a „Küldés” gombra klikkel adatok nélkül, az űrlap „required” feltétele miatt jelzi a böngésző, hogy az első dátumot kötelező megadni
* A képen szöveg látható

  Automatikusan generált leírásha két dátumot adunk meg, de a kezdő dátum későbbi napot jelöl, mint a záró dátum, akkor hibaüzenettel jelzi a felhasználónak

13. ábra: Hibaüzenet kérelem leadásakor

* ha a kezdő dátum és záró dátum feltételei megfelelnek, akkor más hibát nem jelez, az időintervallum bármekkora lehet. Ez még ha nem is terheli túl a rendszert, az adatbázisba és így a főnök felhasználói felületére túl sok adatot küld
* ha megfelelően állított be minden a felhasználó, akkor sikeresnek könyveli el az oldal a kérelmet

Értesítések (főnök felületen):

* ha egyetlen jelölőnégyzetet sem választ ki a felhasználó, akkor hibaüzenettel figyelmezteti az oldal, mivel nincs adat, amit elküldhetne
* A képen asztal látható

  Automatikusan generált leírásakár egyesével, akár több nap kijelölésével is sikeres a jóváhagyás.

14. ábra: Hibaüzenet a szabadság elfogadásoknál

Adatok módosítása:

* a jelszó módosítás feltétele, hogy meg kell adni az aktuális jelszót. Ha ez nem történik meg, vagy hibásan adja meg a felhasználó, akkor nem lehetséges új jelszó megadása
* szintén hibát jelez, ha a felhasználó nem ad meg új jelszót, azaz üresen hagyja bármelyik űrlap mezőt
* A képen szöveg látható

  Automatikusan generált leírásA képen szöveg látható

  Automatikusan generált leírásezen kívül feltétel, hogy az új jelszó megadását követően erősítse azt meg ismételt begépeléssel, így akkor sem lehetséges új jelszó rögzítése, ha ez a két jelszó különbözik

15. ábra: Jelszóváltoztatáskor valamelyik mező üres

16. ábra: Jelszóváltoztatáskor helytelen jelszó

## Egységtesztek

A programban szereplő függvények teszteléséhez unit-teszteket is végeztem. Itt nem vizsgáltam, hogy megfelelőek-e a bemeneti értékek, azaz létező felhasználók vagy helyes jelszók kerülnek-e be, hanem hogy a függvények az elvárt módon térnek-e vissza adatokkal. Ezek a tesztek *a funkciokTest.php* fájlban találhatók, és megvizsgálják, hogy megfelelően működik-e:

* a felhasználói jogot eldöntő függvény,
* a jelszókat összevető több függvény,
* az SQL-parancsokat összerakó algoritmus, mint a szabadság és táppénz kérelem
* a hónap nevét visszaadó függvény.

A képen szöveg, képernyőkép, képernyő látható

Automatikusan generált leírásA képen szöveg látható

Automatikusan generált leírás

17. ábra: Sikeres tesztek

18. ábra: Unit-tesztek

# Továbbfejlesztési lehetőségek

Az ötletek gyűjtése során számtalan funkció merült fel bennünk, amelyekből válogattunk, részben mert nem tartottunk szükségesnek, részben pedig a megvalósítás tűnt összetettnek. Mivel már létező programokból merítettünk inspirációt, amelyek mégsem voltak valamilyen okból szimpatikusak, ezért továbbfejlesztésre alkalmasnak gondolnám a feladatot.

Továbbfejlesztésként hasznos funkció lenne, ha az osztályvezetők maguk adhatnák meg, hogy a dolgozóik mely napokon, és milyen időintervallumban dolgoznak. A jelenlegi verzió egy általános munkarendet követ, de sok esetben hasznos lenne, ha több munkarendet meg lehetne adni, a legjobb megoldás pedig az lenne, ha teljesen szabadon lehetne az időpontokat kezelni (például egy ételfutárokat irányító rendszerben, ahol mindenki annyi munkát vállal, amennyit tud).

A funkció, amit hozzátennék, az egy kapcsolat az adatbázissal, ami segít eltárolni a dolgozók előző havi munkarendjét, ami így a könyvelésnek segítene az elszámolásban. Ehhez szükség lenne olyan mezőre, amely többféle értéket tud rögzíteni, nem csak szabadság és táppénz értéket vehet fel.

Ezeken kívül voltak olyan célkitűzések, amelyek benne voltak a tervezetben, de nem úgy sikerült megvalósítani, ahogy szerettem volna:

* a NAPTÁR oldal helyett BEOSZTÁSAIM oldal létrehozása lett volna a cél, ahol nem csak a munkanapokat látja a dolgozó, hanem kattintással tud azon változtatni. Ehelyett jött létre a KÉRÉSEK oldal, és egy űrlap végzi el ezt a funkciót
* JELENLÉTIK oldal, amelyben visszanézhetőek az előző havi munkarendek, kiegészítve a szabadságokkal, táppénzekkel
* az oldal reszponzívvá alakítása, amely csak részben valósult meg, asztali számítógépen és laptopon kisebb méretűvé alakítva megközelítőleg teljesültek az elvárásaim, de az okostelefon kijelzőjén történő megjelenítés még további fejlesztés igényelne.

# Felhasználói dokumentáció

A MikorDolgozom nevű weboldal egy munkahelyi beosztás-kezelő oldal, ahol az egy céghez tartozó dolgozók tekinthetik meg munkabeosztásukat, igényelhetnek szabadnapot, és jelezhetik, ha betegszabadságon vannak. Az osztályvezetők pedig megtekinthetik a csoportjukba tartozó alkalmazottakat és jóváhagyhatják az általuk igényelt szabadság-kérelmeket.

Az oldal funkciói csak bejelentkezés után érhetőek el, amihez a cég adja meg a belépési kódot. Ez a dolgozók által előzetesen megadott e-mail cím, valamint egy alapértelmezett jelszó (1234) által történik. A felhasználók adatait a rendszergazda rögzíti az adatbázisban, mint az e-mail címet, személyes adatokat, a rendelkezésre álló szabadnapokat.

## Hardver és szoftver követelmények

A weboldal használatához egy internet kapcsolattal rendelkező eszközre van szükség, mint egy számítógépre, laptopra, tabletre vagy okostelefonra. Ezen kívül egy feltelepített böngészőt igényel, mint a Firefox, Google Chrome vagy Microsoft Edge, valamint internet hozzáférést.

## A program telepítésének, konfigurálásának leírása

Mivel egy webes felületről van szó, a program használata nem igényel telepítést, a böngészőben való megnyitás után szabadon használható.

## A program használatának részletes leírása

### Bejelentkezés

A bejelentkező oldalon a weboldal rövid bemutatása látható, majd egy űrlap, ahol a felhasználó e-mail címét és jelszavát kell megadni és a ’Belépés’ gombra klikkelni. Enélkül csak a bejelentkező felület látható, idegeneknek nincs lehetősége online regisztrálni. Sikeres belépés esetén jelenik meg a menüsor az elérhető funkciókkal, és az oldalsávban a felhasználó személyre szóló üdvözlése.

Az első bejelentkezést követően érdemes megváltoztatni a jelszót biztonsági okokból, mert mindenkinek alapértelmezetten ’1234’ a jelszava.

### Naptár

19. ábra: Bejelentkező oldal

A NAPTÁR menüpontra klikkelve megjelenik a 2022-es évre vonatkozó naptár, különböző színekkel jelölve a munkanapokat, hétvégéket és munkaszüneteket. Látszódnak a hét napjai, hogy melyik dátum milyen napra esik. A munkanapok zöld színnel jelennek meg, a hétvégék sárgával, az ünnepek pedig pirossal. Ha a megszokottól eltérő munkarendű nap fölé húzzuk az egeret (pl. ünnepnap), akkor megjelenik a naphoz tartozó leírás. Ezen az oldalon csak megtekintésre van lehetőség, szerkesztésre nem.

### Kérések oldal

A képen szöveg látható

Automatikusan generált leírásEzen az oldalon van lehetőség szabadságot kérelmezni, vagy jelezni a munkáltató felé, hogy táppénzen van a felhasználó. A kezdő dátum megadása kötelező, ha csak egy dátumot jelöl meg, akkor egy napra küld kérelmet, ha pedig megadja a záró dátumot is, akkor a két dátum közé eső összes napra (beleértve a kezdő és záró dátumot is) szól a kérelem. A dátumok alatt két rádió gomb jelenik meg, „Szabadság” és „Táppénz” feliratokkal, amelyek közül egyet kötelező választani (alapértelmezetten a „szabadság” van kiválasztva). A „Küldés” gomb megnyomása után a szabadság igény a főnökhöz kerül, akinek a jóváhagyása szükséges, táppénz esetén automatikus az elfogadás.

20. ábra: Kérelem űrlap

### Értesítések

A képen asztal látható

Automatikusan generált leírásAz ÉRTESÍTÉSEK fülre kattintva a felhasználói minőségtől függően más adatok jelennek meg. Beosztottként a vezető által elfogadott szabadság dátumait láthatja legfelül, ha még egy sem került elfogadásra, akkor a „Nincsen elfogadott kérelem.” felirat jelenik meg. Alatta az elfogadásra váró kérelmek látszanak, ha nincs mit megjeleníteni, akkor a „Nincsen elfogadásra váró kérelem.” üzenet látható. Legalul pedig a megmaradt szabadnapjai száma jelenik meg.

21. ábra: Alkalmazott értesítései

A képen asztal látható

Automatikusan generált leírásFőnökként a beérkezett szabadság kérelmeket látja a felhasználó egy táblázat formájában. Itt megjelenik a beosztottja neve és a dátum, amelyre a szabadságot igényli, a sorok mellett pedig egy jelölőnégyzet. Ezt bejelölve és a „Küldés” gombra kattintva elfogadottá válik az adott napra a kérelem.

22. ábra: Szabadság elfogadása

### Beosztottjaim oldal

Csak vezetőként látható oldal, ahol az osztályához tartozó személyek, beosztottak listája található. Szerkesztésre nincs lehetőség, csak megtekintésre.

### Személyes

A „SZEMÉLYES” fül fölé húzva az egérmutatót, megjelenik egy legördülő menü, ahol a „SZEMÉLYES ADATOK MÓDOSÍTÁSA” és a „KIJELENTKEZÉS” lehetőségek közül lehet választani.

A „Személyes adatok módosítása” oldalon van lehetőség új jelszót megadni, ehhez helyesen kell beírni az addig használt jelszót, majd kétszer megadni az újat, amire változtatni szeretné a felhasználó. Az új jelszónak és a megerősítő jelszónak egyeznie kell.

A „Kijelentkezés” menüpontot választva kilép a személyes profiljából és visszakerül a Főoldalra a bejelentkezési felülettel.

# Összegzés

A vizsgaremek elkészítése eddig messze a legnagyobb feladat volt, amivel szembesültem a kétéves képzés során. Már maga a tervezés folyamata is kihívás volt, az embernek számtalan elképzelése van arról, hogy mit szeretne létrehozni, mi az, ami a követelmény, és mi a képességei alapján megvalósítható.

Nem sikerült mindent a tervek szerint megvalósítani, és mindig van, amit az ember szeretne továbbfejleszteni és csiszolni, de úgy érzem, hogy ezek mind pótolható részek, csupán a rendelkezésre álló idő nem tette lehetővé a tökéletesítést.

Már a tervezés is nehéz folyamat volt, ami részben a projektmunkának is köszönhető. A téma kitalálásától a részletek kidolgozásáig mindenben konszenzusra kellett jutnunk, ami hosszabbá tette ezt a szakaszt, mintha egy meghatározott feladat lett volna megadva konkrét lépésekkel. Ez ugyanakkor lehetőséget teremtett az egymástól való tanulásra.

Nehézség volt még a megvalósításnál, hogy a tanultakat megfelelően alkalmazzam a feladat megoldása során: minden funkció elkészítése egyedi, nem lehet egy sablont követni, így az is kihívás volt, hogy el tudjak vonatkoztatni a konkrét példáktól, és a megoldandó problémára szabjam a kivitelezést.

Minden fennakadás ellenére azonban pozitív élményt nyújtott a projekt elkészítése, mert megerősített abban, hogy kitartással minden feladat megoldható.

# Irodalomjegyzék, forrásjegyzék

<https://developer.mozilla.org/en-US/>

<https://www.w3schools.com/>

<https://getbootstrap.com/>

<https://www.php.net/manual/en/index.php>

<https://prog.hu/hirek/4808/99-9999-ban-torhetok-az-sha-1-gyel-kodolt-jelszavak>

1. Bár a jelenlegi álláspont szerint az SHA1 hashelési algoritmus már nem tekinthető biztonságosnak, az általunk használt adatbázis-kezelő felületen a helyette ajánlott bcrypt vagy SHA512 hash-függvények nem elérhetők, így ezt a megoldást választottuk. [↑](#footnote-ref-1)