



Középiskolai e-napló fejlesztése MVC keretrendszer használatával

Készítette

Renyhárt Gábor

programtervező informatikus BSc

Témavezető

Balla Tamás

tanársegéd

EGER, 2020

Tartalomjegyzék

1. Probléma kifejtése	4
2. Implementálás	5
2.1. Codeigniter	5
2.2. Javascript	6
2.3. Bootstrap	6
2.4. Sublime Text 3	7
2.5. WAMP	7
2.6. Az adatbázis	7
3. Problémák, és megoldásuk a fejlesztés során	9
4. Felhasználói dokumentáció	10
5. Továbbfejlesztési lehetőségek	11
5.1. Külső fájl importálása	11
5.2. Órarend	11
5.3. Házi feladatok	12
5.4. Szülői, orvosi igazolás feltöltése	12
5.5. Üzenet státusza	12
6. Összefoglalás	13

Bevezetés

Mindannyian voltunk diákok, jártunk iskolába, és mindenki kapott jegyet. Az én időmben még papír alapon. A naplóba, amitől mindig féltünk feleléskor, hogy hol nyílik ki. Már általános iskola végén sem értettem, hogy ha ennyi számítógép van már a világban, akkor miért nem használják egy ilyen területen, ahol sok az adminisztráció, és sok a hibázási lehetőség.

Erre lett megoldás Magyarországon a 2018 szeptemberében kötelezővé tett e-napló használata az iskolákban. Korábban már több intézmény használt saját fejlesztésű, vagy egy vállalat által elkészített elektronikus naplót.

A Digitnaplo Kft. által fejlesztett IEAR (Iskolai Elektronikus Adminisztrációs Rendszer) felkerült az Educatio Nonprofit Kft által összeállított azon programok listájára, melyeket az oktatási intézmények hivatalosan használhatnak (2016 szeptember).

A Mozaik Kiadó által kifejlesztett elektronikus napló, a mozaNapló lehetővé teszi az iskola mindennapjai során felmerülő adatkezelési, szervezési és statisztikai feladatok számítógéppel történő elvégzését. A digitális napló alkalmazásával feleslegessé válik a hagyományos papíralapú naplók vezetése és jelentősen csökkenti a pedagógusok adminisztrációs terheit. Minden internetkapcsolattal rendelkező számítógépről elérhető, nem szükséges külön programot telepíteni. Használatához mindössze alapfokú internetkezelési ismeretek szükségesek.

A Neptun KRÉTA (Köznevelési Regisztrációs és Tanulmányi Alaprendszer) a továbbiakban KRÉTA a köznevelési intézmények oktatásszervezői feladatait támogató informatikai rendszer, amely a köznevelés más rendszereivel integráltan és adaptívan együttműködik.

Az évek során a tanulmányaim folytán jutottam el arra a szintre, hogy képes vagyok egy ilyen alkalmazás megírására. A programom elkészítése során megvizsgáltam a fentebb említett különböző rendszerek működését, funkcióit, előnyeit. Olvastam véleményeket, és ez alapján állítottam össze a saját követeleménylistámat az én e-naplómmal kapcsolatban. Ki tudja, esetleg pont ez az alkalmazás lesz a jövő középiskolai elektronikus rendszerének az alapja.

1. fejezet

Probléma kifejtése

A jelenleg használatban lévő e-naplók a vélemények szerint bonyolultak, sok időt vesznek el az oktatóktól, az adminisztrátoroktól, és a diákoknak sem esik kézre. Viszont nincsenek olyannyira kihasználva, amennyire a XXI. századi felhasználóknak szüksége lenne rá. A célom, hogy egy olyan alkalmazást készítsek el, ami kellően egyszerűen, gyorsan használható és szinte tökéletes. Csak szinte, hiszen mindig lesznek olyan dolgok, amiket egyszerűbben nem lehet megoldani, csak ha máshonnan veszünk el erőforrást. Azaz ha egy funkció egyszerűbb a felhasználónak, az a feldolgozó rendszernek bizonyosan bonyolultabb. Tovább tarthat az adatok kinyerése az adatbázisból, valamint a megjelenítés is tovább tarthat, ami több – azaz néhány száz vagy akár ezer felhasználónál – már jelentős lassúsággal járna.

A fejlesztés során folyamatosan konzultáltam tanár szakos hallgatókkal, akik ismerik a jelenlegi helyzetet, és egyes dolgokat meg tudnak közelíteni más módokon is. Több tíz éve a pályán lévő tapasztalt pedagógusok véleményét is kikérdeztem, hiszen ők a régebbi papír alapú rendszert is használták, és ezt az új e-rendszerben is benne vannak.

2. fejezet

Implementálás

A programot PHP nyelven írtam, a Codeigniter keretrendszer segítségével. Néhány helyen alkalmaztam Javascriptet a felhasználói élmény fokozása érdekében. A megjelenítésért pedig a Bootstrap a felelős. A kódot a Sublime Text 3 programmal írtam. A saját rendszeremen a WAMP alkalmazás szolgáltatásait használtam.

2.1. Codeigniter

A Codeigniter használata egyszerű, könnyen lehet teljes értékű web alkalmazásokat fejleszteni vele. A jelenlegi stabil kiadása a 4.0.4 verzió, amit 2020. július 15-én adtak ki. A szakdolgozati programom elkészítésének a kezdetén még a 3-as verzió volt a legújabb, így azzal a verzióval haladtam a végéig. A Codeigniter egy nagyon hatékony, mégis kevés memóriát és lemezterületet használó PHP keretrendszer. Az MVC tervezési mintát használja, amelynek lényege, hogy a sok adatot a felhasználó elé táró számítógépes alkalmazásokban az adathoz (modell) és a felhasználói felülethez (nézet) tartozó dolgok szétválasztódjanak azért, hogy a felhasználói felület ne befolyásolja az adatkezelést, és az adatok átszervezhetők legyenek a felhasználói felület változtatása nélkül. Ezt az elképzelést a Codeigniter egy controller közbeiktatásával éri el.

Működése a következő:

Egy weboldal betöltése a megfelelő controller megfelelő metódusával kezdődik.

```
1 class Welcome extends CI_Controller
2 {
3     public function index()
4     {
5         $data['news'] = $this->news_model->get_news();
6         $this->load->view('news/index', $data);
7     }
8 }
```

Ez a programrészlet hozzáfér az adatbázishoz a modellen keresztül. A modellben valamilyen adatbáziskezelői parancs fut le. Úgymint lekérdezés, beszúrás, törlés, módosítás.

```
1 class News_model extends CI_Model
2 {
3     public function get_news()
4     {
5         $query = $this->db->get('news');
6         return $query->result_array();
7     }
8 }
```

A parancs lefutása után a vezérlés – lekérdezés esetében a lekérdezett adattal együtt – a controllerbe kerül vissza, amely előkészíti az adatokat a view számára, majd meghívja a viewt, ami a 7.sor alapján a news/index helyen található, majd a \$data változón keresztül hozzákapcsolja a szükséges adatokat. Amint a view megkapta az adatokat, megjeleníti azokat, és kezdődik minden előről, a felhasználó interakciója alapján.

2.2. Javascript

A Javascriptet 1995-ben hozták létre, az akkori statikus weboldalak fejlesztéséért. Több paradigmás nyelvként a Javascript támogatja az eseményvezérelt, a funkcionális és az imperatív programozási stílusokat. Alkalmazásprogramozási interfészekkel (API) rendelkezik a szöveggel, dátummal, reguláris kifejezésekkel, adatstruktúrákkal való munkához. Magának a nyelvnek nincs be-kiírató funkciója, ezeket a feladatokat a gazda környezet, általában egy web böngésző biztosítja.

2.3. Bootstrap

A Bootstrap egy nyílt forrású webes keretrendszer, ami az informatív weboldalak fejlesztésének leegyszerűsítésére fókuszál. Az elsődleges célja, hogy ezt egy webprojekthez adjuk, az nem más, mint megkaphassuk a Bootstrap választásait a színek, méretek, betűtípusok, és kinézetek tekintetében. Amint egy projekthez hozzáadjuk, a Bootstrap biztosítja az alap stílus meghatározásokat minden HTML elem számára. Az eredmény egy egységes megjelenés a táblák, és űrlap elemek megjelenésének minden webböngészőben. A legelső változatot 2011. augusztus 19-én adták ki, miután átnevezték a projektet Twitter Blueprint-ről. A Bootstrap 2 2012 januárjában jelent meg, a 3-as verzió pedig 2013 nyarán. A 4. kiadást 2018 elején véglegesítették. A Bootstrap 5 Alfa verziója 2020 nyarán jelent meg.

2.4. Sublime Text 3

A Sublime Text egy shareware többplatformú forráskód szerkesztő. Számos programozási és jelölő nyelvet támogat. Pluginek segítségével a felhasználók funkciókat adhatnak hozzá.

2.5. WAMP

A WAMP egy Windowson futtatható alkalmazáskiszolgáló programcsomag, amely fő elemeit az alábbi négy program alkotja:

- Windows, a Microsoft által gyártott operációs rendszer.
- Apache HTTP Server, egy szabad szoftver/nyílt forrású webszerver, jelenleg a legnépszerűbb.
- MySQL, egy többszálú, többfelhasználós SQL adatbázis-kezelő rendszer (DBMS), a Sun Microsystems tulajdonában, több mint 11 millió installációval.
- PHP (PHP: Hypertext Preprocessor), egy programozási nyelv, amit eredetileg dinamikus weboldalak fejlesztésére terveztek. A PHP-t leggyakrabban szerveroldali alkalmazásoknál használják, de parancssorból/konzol alól is használható.

2.6. Az adatbázis

Az adatbázisban az adattáblák harmadik normálformában vannak, 19 tábla összekapcsolásából áll össze az alkalmazás. Néhány tábla szerkezete:

users: A program felhasználóinak adatait tárolja. A személyes adatai mellett a diákok esetében az osztályát, és minden felhasználónak a beosztását.

jegyek: A diákok jegyeit tartalmazza. A tábla tartalmazza természetesen a diák azonosítóját, a jegy adóját, azaz a tanár azonosítóját. Az jegy adásának időpontját másodperc pontossággal, magát a jegyet, és hogy melyik tárgyból kapta az érdemjegyet. Két mező van arra, hogy egy jegyhez tartozó dolgozatot lehessen hozzáfűzni, és egy van arra, hogy a diák dolgozatát lehessen ide feltölteni.

haladasi_naplo: A tanárok ebbe a táblába töltik fel azt, hogy melyik órán mit csinálnak, illetve mit fognak csinálni. A tantárgy azonosítóját természetesen tartalmazza, emellett az adott tantárgy, adott osztályában lévő óra sorszámát, az aznapi óra dátumát, és természetesen a tevékenységet, hogy azon a napon mi történik.

users_sets: A felhasználók által választható az oldal megjelenítése bizonyos funkcióknál. Ebben a táblában ez kerül eltárolásra. A felhasználó azonosítója természetesen

itt van, aztán a `set_name` mezőben tárolom azt, hogy melyik kinézetről van szó. Például az 'ofonezet' beállítás az osztályfőnöknek azt a beállítását tartalmazza, hogy a saját osztályának a jegyeit vagy a hiányzását jelenítse meg. Maga a beállítás értéke pedig -1-et, illetve 1-et vehet fel. A program ez alapján eldönti, hogy mely kinézetet kell megjelenítenie.

szulogyermek: Ebben a táblában kerülnek összekapcsolásra a szülők a gyermekeikkel. El van tárolva a szülő azonosítója, a gyermek azonosítója, valamint egy harmadik mezőben, amelynek neve: 'hasznalt', annak a gyermeknek az azonosítója szerepel, akit a szülő jelenleg megtekinteni kíván. A többi gyermekénél ez a mező 0-t tartalmaz.

szuloid	gyermekid	hasznalt
23	21	0
23	22	22

2.1. ábra. A 'szulogyermek' tábla részlete

uzenetek: Ez a tábla felelős az üzenetek tárolásáért. Tartalmazza természetesen a feladó és a címzett azonosítóját, valamint egy 'csoport' nevű mezőt is, mely a feladó és a címzett adataiból áll össze. Erre a jobb kereshetőség miatt volt szükség. Az üzenet dátumát, valamint természetesen az üzenet szövegét is tartalmazza. Tárgy mezőt nem hoztam létre, mert gondolataim szerint ez inkább egy 'Az iskola és a szülő közleményei' felület lenne, amelyben a szövegből egyértelműen kiderül az üzenet tárgya. Minden üzenetnek van egy státusza, ezt a státuszt már sajnos nem volt időm kidolgozni, de az értelme az lenne, hogy ha még nem olvasta a címzett, akkor a megjelenítésben jobban ki lenne emelve, jelezvén, hogy ez még nem lett elolvasva.

3. fejezet

Problémák, és megoldásuk a fejlesztés során

Ebben a fejezetben kifejtem, hogy az alkalmazás mely részénél futottam bele olyan problémákba, amit nem tudtam egyszerűen, könnyen átlépni. Először a probléma felmerülésének helyét, leírását adom meg, majd több megoldást kipróbálok, végül leírom miért pont az adott megoldás lett a tökéletes.

4. fejezet

Felhasználói dokumentáció

Ebben a fejezetben a felhasználók számára, világosan, érthetően, képernyőképekkel illusztrálva írom le a program működésének minden olyan funkcióját, amire szükségük lehet az alkalmazás használata során.

5. fejezet

Továbbfejlesztési lehetőségek

Szeretnék egy olyan rendszert megalkotni, ami már-már továbbfejleszthetetlen. De természetesen ilyen nem létezik, soha nem is fog. Minden programban vannak továbbfejlesztési lehetőségek.

5.1. Külső fájl importálása

Akár Excel, akár csv fájl is fel lehetne tölteni, amit a program feldolgoz, és az adatokat a megfelelő helyekre tárolja el. Például ha egy diák év közben érkezik az iskolánkba, akkor a személyes adatait felvehetjük, nem tart sokáig, ám az előző intézményben szerzett korábbi jegyeire is szükségünk lenne. Ezeket egyesével nagyon sok idő lenne felvinni, és a hibázási lehetőség is nagyon nagy. Ekkor jön a következő gondolat, hogy akkor Exportálásra is lenne igény, így tényleg a diák összes adatát egy pillanat alatt lementhetjük, és a másik rendszerben pedig beimportálás után használhatjuk.

Természetesen minden év elején a sok új diákot, és a szüleit is könnyedén fel lehetne tölteni a rendszerbe, amennyiben egy megfelelően formázott dokumentum rendelkezésre áll a feltöltésre.

5.2. Órarend

Egy elektronikus naplóban kézenfekvő megoldás, ha az órarendet is már az alkalmazás készíti el. Ahhoz, hogy ezt el lehessen készíteni, szükség van a tanárok, az osztályok, és a tantermek rögzítésére. Természetesen nem megfelelően a tantárgyakról, ami minden osztály esetében más és más. Ezek az adatok az általam készített programokban természetesen megvannak, így már csak maga az órarend generálásra lenne hátra. Ennek a megalkotása viszont felérne egy újabb fél éves fejlesztési folyamattal.

5.3. Házi feladatok

Ugyanúgy, ahogyan a jelenlegi programban a dolgozatok meg vannak valósítva, meg lehet ugyanezt a házi feladatra is valósítani. A tanár kiadja a feladatot online, és a diákok otthon megcsinálva, feltölthetik a naplóba, ahonnan a tanár letöltés, nyomtatás, és piros toll használata nélkül egyszerűen tudná ellenőrizni.

5.4. Szülői, orvosi igazolás feltöltése

Megvalósítható az is, hogy a szülő maga írja meg a Szülői igazolást, amit elmenthet Igazolásként. Ezt az osztályfőnök látja, és ez alapján állítja a meglévő hiányzás státuszát a megfelelőre. Az orvosi igazolással hasonló a helyzet. Amint a diák vagy a szülő kézhez kapta a betegséget igazoló dokumentumot, akkor feltöltheti azt képként a rendszerbe.

5.5. Üzenet státusza

Ahogy korábban már leírtam, időm nem volt rá sajnos, de a felhasználók segítségére válna egy olyan funkció, hogy a beérkezett üzeneteket megjelölhetné az alkalmazás valami feltűnő módon, hogy ez az üzenet, bizony még nem lett elolvasva. Gondolok itt akár kerettel körbevenni, vagy az üzenet elejét félkövér betűkkel kiemelni. Akár plusz gombbal tényleges elolvasás nélkül is lehetne olvasottnak jelölni.

6. fejezet

Összefoglalás

Ebbe a fejezetbe kerül a probléma ismételt leírása, valamint a megoldásával együtt egy következtetés levonás.

Irodalomjegyzék

<https://www.digitnaplo.hu/>

<https://www.mozaik.info.hu/Homepage/Mozaportal/MPdigitalis.php?op=mozanaplo>

<https://ekreta.hu/>

<https://hu.wikipedia.org/wiki/WAMP>

<https://matebalazs.hu/javascript.html>

<https://thispersondoesnotexist.com/>

<https://codingislove.com/realtime-search-javascript/>

http://www.sulibolt.hu/asc_ism.html