Matematikai és Informatikai Intézet

Középiskolai e-napló fejlesztése MVC keretrendszer használatával

Készítette
Renyhárt Gábor
programtervező informatikus BSc

Témavezető
Balla Tamás
tanársegéd

Eger, 2020

Tartalomjegyzék

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 követeléménylistámat 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.

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 elölről, 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-kiíratója funkciója, ezeket a feladatokat a gazda környezet, általában egy web böngésző biztosítja.

2.3. Bootstrap

A forráskódot a Sublime Text 3 programmal írtam. A saját számítógépemen a WAMP programot használtam a szerver funkciójának betöltésére. A WAMP négy program nevéből alkotott szó:

- Windows, a Microsoft által gyártott operációs rendszer.
- Apache HTTP Server, egy szabad szoftver/nyílt forrású webszerver, jelenleg a legnépszerűbb.
- MySQL, egy többszálas, 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.

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

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 fájlt 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 megfeledkezve 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.

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