

SZAKDOLGOZAT



MISKOLCI EGYETEM

Információs weboldal informatikus hallgatók számára Laravel keretrendszerben

Készítette:

Schimkó Nelly Gabriella

Programtervező informatikus szak

Témavezető:

Dr. Baksáné Dr. Varga Erika

MISKOLC, 2018

SZAKDOLGOZAT FELADAT

Schimkó Nelly Gabriella (FOFX80) programtervező informatikus jelölt részére.

A szakdolgozat tárgyköre: Modern webtechnológiák

A szakdolgozat címe: Információs weboldal informatikus hallgatók számára Laravel keretrendszerben

A feladat részletezése:

Rövid elméleti összefoglaló a programhoz szükséges technológiákról, illetve a webes szolgáltatásokról.

A hallgatók számára szükséges információk megszerzése, felhasználók igényeinek megismerése, felmérése.

Webes hallgatói információs felülettervezése, amely hasznos információkat nyújt a hallgatók számára (egyetemi programok, jegyzetek, munkavállalással kapcsolatos információk), valamint hirdetési felületet biztosít a diákok és a vállalkozások számára (korrepetálás felajánlása, állásajánlatok, nyílt napok, gyárlátogatások).

Az alkalmazás elkészítése design elkészítése, weboldal vázának felépítése, kiszolgáló funkciók és felhasználói oldal elkészítése.

Az alkalmazás működésének tesztelése, alkalmazáshoz tartozó leírás és súgó elkészítése.

Témavezető(k): Dr. Baksáné Dr. Varga Erika Egyetemi docens

Konzulens(ek):

A feladat kiadásának ideje:

.....
szakfelelős

EREDETISÉGI NYILATKOZAT

Alulírott ; Neptun-kód:
a Miskolci Egyetem Gépészmérnöki és Informatikai Karának végzős
szakos hallgatója ezennel büntetőjogi és fegyelmi felelősségem tudatában nyilatkozom
és aláírásommal igazolom, hogy
című szakdolgozatom/diplomatervem saját, önálló munkám; az abban hivatkozott szak-
irodalom felhasználása a forráskezelés szabályai szerint történt.

Tudomásul veszem, hogy szakdolgozat esetén plágiumnak számít:

- szó szerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;
- tartalmi idézet hivatkozás megjelölése nélkül;
- más publikált gondolatainak saját gondolatként való feltüntetése.

Alulírott kijelentem, hogy a plágium fogalmát megismertem, és tudomásul veszem,
hogy plágium esetén szakdolgozatom visszautasításra kerül.

Miskolc, év hó nap

.....

Hallgató

1.

szükséges (módosítás külön lapon)

A szakdolgozat feladat módosítása

nem szükséges

.....

dátum

.....

témavezető(k)

2. A feladat kidolgozását ellenőriztem:

témavezető (dátum, aláírás):

konzulens (dátum, aláírás):

.....

.....

.....

.....

.....

.....

3. A szakdolgozat beadható:

.....

dátum

.....

témavezető(k)

4. A szakdolgozat szövegoldalt

..... program protokollt (listát, felhasználói leírást)

..... elektronikus adathordozót (részletezve)

.....

..... egyéb mellékletet (részletezve)

.....

tartalmaz.

.....

dátum

.....

témavezető(k)

5.

bocsátható

A szakdolgozat bírálatra

nem bocsátható

A bíráló neve:

.....

dátum

.....

szakfelelős

6. A szakdolgozat osztályzata

a témavezető javaslata:

a bíráló javaslata:

a szakdolgozat végleges eredménye:

Miskolc,

.....

a Záróvizsga Bizottság Elnöke

Tartalomjegyzék

1. Bevezetés	7
2. A program háttere	8
2.1. Felhasználói célcsoport	8
2.2. Hasonló kezdeményezések	8
2.2.1. Programnyelvek portál	8
2.2.2. BME VIK Wiki	9
2.2.3. Stack Overflow	10
2.3. Saját rendszerrel szembeni követelmények	11
2.3.1. Elérhető funkciók	11
2.3.2. Jövőbeli fejlesztések	12
2.4. Felhasznált technológiák	12
3. Tervezési feladatok	14
3.1. Követelmények	14
3.1.1. Funkcionális követelmények	14
3.1.2. Nem funkcionális követelmények	15
3.2. Adattárolás	16
3.2.1. Adatok szinkronizációja	18
3.2.2. Adatok feltöltése	19
3.3. Felhasználói szerepkörök	20
3.4. Grafikus felület	21
3.4.1. Oldalak kialakítása Laravel használatával	22
3.4.2. Felhasználói élmény növelése az oldalon	23
4. Módszertan	25
4.1. Laravel keretrendszer	25
4.1.1. Rövid története	25
4.1.2. Blade	26
4.1.3. Migrations	26
4.1.4. Controller	27
4.1.5. Route	28
4.1.6. Session	29
4.1.7. Pagination	29
4.1.8. Query Builder	30
4.1.9. Eloquent ORM	30
4.1.10. Authentication	31
4.1.11. Mail	33
4.1.12. HTMLPurifier	34

4.1.13. File Storage	34
4.1.14. Intervention Image	35
4.2. Biztonság és védelem	35
4.2.1. Felhasználói adatok tárolása	35
4.2.2. Jogosultságok beállítása	35
4.2.3. Jelszó visszaállítás	36
4.2.4. Hozzászólások ellenőrzése	38
4.2.5. WYSIWYG szerkesztő ellenőrzése	40
4.2.6. Űrlapok kitöltése	40
5. Tesztelés	41
5.1. Üzenetküldés tesztelése	41
5.2. A készítő általi tesztelés	42
5.3. Felhasználó általi tesztelés	43
6. Összefoglalás	44
6.1. Összefoglalás	44
6.2. Summary	45
Irodalomjegyzék	46
Adathordozó használati útmutató	48

1. fejezet

Bevezetés

A szakdolgozatom elkészítése során egy olyan információs oldalt szerettem volna létrehozni elsősorban az egyetem informatikus hallgatói számára, amely témájában nem csak az egy egyetemi tanulmányaikban, hanem a diákéletben és a pályakezdéssel kapcsolatos kérdéseikben is segítségükre lehet.

A dolgozatom témaválasztásánál két dologban voltam biztos, hogy webfejlesztéssel kapcsolatos irányba szeretnék menni, és, hogy a gyakorlatban is hasznos weboldalt szeretnék létrehozni. A választásom azért efelé a témakör felé terelődött, mert bár láttam több hasznos oldalt ami a diákok segítségére volt, de egyiket se éreztem teljesnek, minden témát átfogónak.

A másik oka az volt hogy ezek a weboldalak nem a Miskolci Egyetemhez kapcsolódtak. Vannak oldalak ahol a programozók kérdéseket tehetnek fel vagy kereshetnek rá, de mennyivel könnyebb konkrét választ kapnunk a kérdésünkre, ha a válaszadó ismeri a feladat egészét, mert ő is ezen dolgozik vagy dolgozott pár éve és hasonló problémákkal találkozott. Persze több csoportot találunk közösségi oldalakon, ami összefog bizonyos évfolyamokat, csoportokat az egyetemen, de nem minden hallgató tagja ezeknek a csoportoknak, illetve nem minden csoportnak tagja, így nem feltétlenül jut el hozzá minden információ.

A dolgozatom célja, egy olyan egységes weboldal elkészítése, amely nagy segítséget jelenthet az egyetem informatikus hallgatóinak számára, abban hogy információkat kapjanak, a tantárgyakkal, diákélettel vagy akár a munka világával kapcsolatban.

A dolgozatban szó lesz a programhoz felhasznált technológiákról, a jelenleg működő rendszerek előnyeiről és hiányosságairól, illetve bemutatásra kerül maga az alkalmazás és annak elkészítési folyamata.

2. fejezet

A program háttere

2.1. Felhasználói célcsoport

Az alkalmazás elsősorban a Miskolci Egyetem programtervező informatikus, mérnök informatikus és gazdasági informatikus hallgatóit célozza meg, de persze más felhasználók számára is remek lehetőséget jelent, akik az informatika világában mozognak vagy valamilyen kapcsolatban állnak vele. Tehát a felhasználók lehetnek:

Diákok: számukra azért kiváló lehetőség ez a weboldal, mert a tartalmakat elrendezve látják, és a weboldal több diákot összefűz évfolyamtól és szakiránytól függetlenül.

Vállalkozások: az oldalra bárki regisztrálhat tehát nem csak diákok, így például vállalkozások, cégek vagy diákszövetkezetek is hirdethetnek az oldalon

Oktatók: természetesen az oktatók is részesei lehetnek az oldalnak, ezáltal közelebb kerülve a hallgatókhoz, könnyebben felmérhetik igényeiket.

2.2. Hasonló kezdeményezések

Több egyetem is működtet hasonló oldalakat, amelyeket a hallgatók gondoznak, töltenek fel információkkal. Ezek az oldalak jó kiindulási pontot jelentenek a felhasználók számára, amennyiben tanulmányaikkal kapcsolatban szeretnének informálódni, azonban a legtöbb ilyen kezdeményezés csak a tananyagok szempontjából nyújt hasznos információt és nem tér ki a diákélet többi területére, mint a programok, szórakozás, pályakezdés.

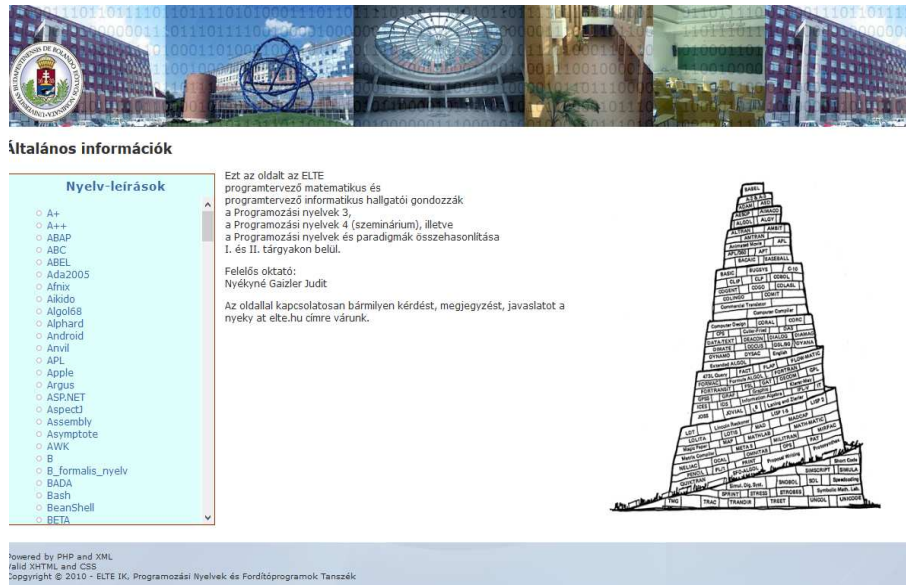
Ebben a részben néhány hasonló oldalt fogok bemutatni, amelyek elemzése a tervezés során nagy segítséget jelentett az igények felmérésének szempontjából, és a rendszertől elvárt követelmények meghatározásában.

2.2.1. Programnyelvek portál

Az ELTE programtervező matematikus és informatikus hallgatói szerkesztik az oldalt négy tantárgyuk keretén belül. Az oldal több programnyelv szintaktikáját ismerteti meg velünk és egyszerű példákkal szemlélteti azok használatát.

ELŐNYÖK: A programnyelvek kategorizálva vannak, illetve a programnyelveken belül is több részre vannak felosztva az oldalak (nyelvi elemek, típusok, vezérlési szerkezetek). A tartalom részletes, bőséges.

HÁTRÁNYOK: Az oldal megjelenésének elrendezése nem legpraktikusabb. A főoldalon egy oldalsávból választhatjuk ki a számunkra szükséges nyelvet, ami egy hosszú lista. Például lehetne tovább csoportosítani a nyelveket az alapján, hogy milyen területen (webfejlesztés, adatbázis kezelés) használjuk fel őket. Ami még problémát okozhat az a megjelenés, mivel néhány szín a szem számára zavaró lehet.



2.1. ábra. A Programnyelvek Portál kezdőlapja

2.2.2. BME VIK Wiki

A VIK Wiki a Budapesti Műszaki Egyetem Villamosmérnöki és Informatikai karának egységes közösségi információs forrása. Az oldal a MediaWiki motorból indul ki, ami a Wikipédiát is hajtja, éppen ezért külső megjelenésében hasonló vonásokat mutat a két oldal.

ELŐNYÖK: Egy nagyon jó csoportosítási rendszer működik az oldalon, karok, tárgyak és félévek szerint is csoportosítva van az oldal. Külön kiemelandő hogy nem csak a tárgyakkal, de az egyetemi közélettel, szakkörökkel is foglalkozik az oldal. Az oldal regisztrációs rendszere a karhoz köthető ezzel is elősegítve az oldal biztonságát.

HÁTRÁNYOK: Az oldal sok információt hordoz magában, de a sok adat között könnyű elveszni. Ezen kívül könnyen kerülhet hibás információ az oldalra, mivel egy Wikipédiáról van szó.

2.2. ábra. A BME VIK Wiki oldala

2.2.3. Stack Overflow

Ugyan nem egyetemhez kapcsolódó oldal, de a diákok gyakran használják feladatik során és a hallgatók számára kifejezetten hasznos egy magyar nyelvű változat, ami kifejezetten kötődik az egyetemi feladatokhoz. A kérdéseken kívül az oldal állásajánlatokat is tartalmaz.

ELŐNYÖK: A válaszok pontozhatóak és ez alapján vannak kilistázva, illetve a kérdést feltevő megjelölheti a számára leghasznosabb választ. Az oldalon tageket (címkéket) használnak ami megkönnyíti a keresést. Az oldalon achievement-rendszer is működik, aminek köszönhetően még több látogatót vonz.

HÁTRÁNYOK: A diákok szempontjából, ami hátránya lehet az igazából előny is, mégpedig hogy nem magyar nyelvű. Segíti a nyelvtanulást, de sokszor jobban értenünk a választ az anyanyelvünkön.

jQuery find() opposite

I want to find the next in a list of many div containers. Find() is a great function to find child objects . But what about the opposite Way to find in parent?

9

```

<form id="grabbMe">
  <div>
    <div>
      <div>
        <div><input type="text" value="test"></div>
      </div>
    </div>
  </div>
</form>

<script>
$('input').findUp('form').attr('id')
</script>

```

jquery find parent

share improve this question edited Sep 14 '15 at 11:00

asked Jan 22 '12 at 13:42 dazzafact 951 10 25

add a comment

2 Answers

active oldest votes

16

You can use jQuery's closest() function to do this.

```

$('input').closest('form').attr('id');

```

share improve this answer answered Jan 22 '12 at 13:44 kfglsang 1,485 1 17 26

add a comment

Search...

0

The opposite of find() is parents() .

closest() isn't quite an opposite to find() . Although it may be what you're looking for. It depends what you're trying to do:

the find() function finds all occurrences of the selector within the descendants of the element you specify.

closest() only finds the first occurrence of the selector, going upward through the ancestors of the specified element.

So the correct opposite to find() would be parents() , which will find all ancestors of your element that match the specified selector.

share answered 2 mins ago Skeets 1,354 13 30

add a comment

2.3. ábra. Egy kérdés a stackoverflow.com oldalról

2.3. Saját rendszerrel szembeni követelmények

2.3.1. Elérhető funkciók

Az oldal bárki számára olvasható, azonban szerkeszteni csak regisztrált felhasználóként lehet őket. Ezt a megfelelő autentikáció kialakításával értem el.

Az oldal több témakörre van felbontva. A felhasználók írhatnak ki híreket, eseményeket, ezekhez feltölthetnek kiemelt képeket, módosíthatják a bejegyzéseiket, illetve törölhetik azokat.

Az oldalra feltölthetőek jegyzetek széleskörű formátumban, illetve ezek letölthetőek bárki számára. Létrehozhatóak projektek vagy kereshetünk az egyetemi feladatainkhoz társakat.

A cégek, iskola szervezetek hirdethetnek az oldalon, megadva a beosztást, a cég nevét és a várost ahol a munkahely található.

A diákok kérdéseket tehetnek fel az oldalon, amelyekre választ kapnak társaiktól esetleg programozóktól.

A hozzászólások törölhetők a bejegyzés írójának számára annak érdekében hogy ezzel növeljem az oldal biztonságát.

2.3.2. Jövőbeli fejlesztések

A jövőben számos fejlesztést tervezek. Az első és legfontosabb, hogy a hozzászólások kedvelhetőek legyenek a kérdések oldalain, annak érdekében hogy a diákok könnyebben válogathassanak a válaszok között, hamarabb találják meg a számukra megfelelő választ és ezáltal gyorsabban haladjanak feladataikban.

A másik ilyen változtatás a felhasználó profil kialakítása, kiegészítése lenne. Az adatok között szerepelne egy felhasználói kép, egy személyes leírás, illetve annak feltüntetése mivel foglalkozik a felhasználó, milyen programnyelveket ismer. Mindezt annak érdekében hogy a felhasználók jobban megismerjék egymást, könnyebben tudjanak esetleg segíteni egymásnak.

A felhasználói élményt tovább növelve javítanám az oldal elrendezését hogy gyorsabban lehessen bejegyzéseket kiírni, illetve javítanék az oldal reszponzivitásán, hogy minden eszközön jól olvasható legyen.

Továbbá létrehoznék egy külön oldalt a tantárgyak számára, arról a diákok hasznos leírásokat találnak az egyes tárgyakról, illetve a jegyzeteket is kategóriák szerint csoportosítanám.

Egyes pontokon még hiányzik a magyarázás, ezt pótolnám. Illetve, ha nyilvánossá tenném az oldalt változtatnék a nevén.

2.4. Felhasznált technológiák

LARAVEL

Egy nyílt forráskódú PHP keretrendszer, amely az MVC (Model-Model, View-Nézet, Controller-Vezérlő) szoftverfejlesztési mintát követi. Előnye, hogy egy csomagban ötvöz számos szolgáltatást, mint a RESTful routing (útvonalkezelés), könnyű e-mail küldés, adatbázis migrációk kezelése.

Jellemzője, hogy egyszerű kifejező szintaxist használ és rengeteg gyakran ismétlődő feladat megoldását kínálja fel.

BOOTSTRAP

Ez egy HTML, CSS, JavaScript alapú keretrendszer. Sok előredefiniált komponenst tartalmaz pl.: gombok, tabok, menük, lenyíló menük. Nagyban megkönnyíti a reszponzív weboldalak létrehozását, mivel alapja a 12 rácsos megjelenés. Megkönnyíti a munkát illetve egyre gyakrabban használt, lényegében már minden sablon elkészítésénél használják Bootstrapot.

MYSQL

A MySQL egy többfelhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerver.

A szoftver eredeti fejlesztője a svéd MySQL AB cég, amely kettős licenccel tette elérhetővé a MySQL-t; választható módon vagy a GPL szabad szoftver licenc, vagy egy zárt (tulajdonosi) licenc érvényes a felhasználásra. 2008 januárjában a Sun felvásárolta 800 millió dollárért a céget. 2010. január 27-én a Sun felvásárolta az Oracle Corporation, így a MySQL is az Oracle tulajdonába került.

A MySQL az egyik legelterjedtebb adatbázis-kezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú LAMP (Linux–Apache–MySQL–PHP) összeállítás részeként költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására.

HTML

Leíró nyelv, amelyet weboldalak készítéséhez fejlesztettek ki. A HTML a weboldalak szerkezetét írja le egy jelölőnyelv által. Jelenleg a HTML5-ös verzióját használjuk.

CSS

Egy stílusleíró nyelv, amely a HTML dokumentum megjelenését írja le. Célja, hogy munkát takarítsunk meg vele ugyanis egy CSS fájl több oldalhoz is használható.

PHP

Általános szerveroldali szkriptnyelv dinamikus weboldalak készítéséhez. Az első skriptnyelvek egyike, amely HTML fájlba ágyazható külső fájl helyett. A kódot maga a webszeres értelmezi.

A PHP nyílt forráskódú szabad felhasználású szoftver, manapság inkább valamilyen keretrendszerbe (Laravel, Symfony, PHP Yii) használjuk.

SQL

Relációs adatbázis-kezelők lekérdezési nyelve. A relációs adatbázis-kezelők SQL segítségével programozhatóak. Az SQL nyelvi elemeket 4 részre lehet osztani: adatdefiníciós (Data Definition Language, DDL), adatkezelési (Data Manipulation Language, DML), lekérdező (QUERY(Language - QL)) és adatvezérlő (Data Control Language, DCL).

JQUERY

Egy könnyűsúlyú JavaScript könyvtár amelynek célja, hogy könnyebbé tegye a JavaScript nyelv felhasználását a weben. Amíg JavaScript-el többsornyi kódot kell leírunk, addig a jQuery ezeket a módszereket lényegében egy sorba tömöríti. A jQuery megkönnyíti a DOM és az AJAX használatát.

3. fejezet

Tervezési feladatok

Ebben a fejezetben leírom a szakdolgozatom első felében kialakított terveket, azok változásait, illetve hogy végül miként valósítottam meg őket.

3.1. Követelmények

3.1.1. Funkcionális követelmények

- Az oldal mindenki számára legyen olvasható: a megvalósult weboldal olvasásához nincs szükség regisztrációra, azonban már a posztokhoz való hozzászólás is regisztrációhoz kötött.
- Az adminisztrátor ellenőrizhesse a hozzászólásokat és törölhesse azokat amennyiben azok tartalma nem megfelelő és nem hasznos a közösség számára: előfordulhat hogy valaki a hozzászólásával megsérthet más csoportokat, személyeket, ennek kiküszöbölésére hasznos megoldás, hogy a felhasználók látják a bejegyzésükre érkezett hozzászólásokat amiket ezután lehetőségük van törölni.
- Az oldalon minden regisztrált felhasználó létre tudjon hozni bejegyzéseket: a bejegyzések létrehozása és szerkesztése is regisztrációhoz kötött, az admin oldalon minden felhasználó csak a saját maga által létrehozott bejegyzéseit látja ezeket módosíthatja és törölheti.
- A bejegyzések legyenek kategóriákba sorolva: a bejegyzések csoportosítva vannak hírek, projektek, kérdések, állások és jegyzetek kategóriában.
- A főoldalon az egyes kategóriáknak csak a legnépszerűbb bejegyzései jelenjenek meg: ezt végül nem valósítottam meg és nem is valószínű hogy szükséges, hiszen egy kategórián belül nem fog összegyűlni annyi bejegyzés, ami ezt szükségessé teszi, a későbbiekben a kérdések oldalon szeretném a legtöbb választ kapott kapott kérdések alapján is listázni a bejegyzéseket.
- A regisztrált felhasználók számára legyen lehetőség a hozzászólásra: a továbbiakban szeretném hogy nem csak kommentelhetőek legyen az oldalak, de pontozni vagyis kedvelni is lehessen a hozzászólásokat.
- A fórumon a kérdésekre adott válaszokat lehessen értékelni annak hasznossága szerint: ez a funkció jelenleg nem érhető el.

- A kezdő oldalon legyen egy slider, ami néhány kiemelt bejegyzésre mutasson.
- Minden bejegyzéshez lehessen kiemelt képet rendelni: minden kategóriában megvalósul ez a funkció amelyikben szükséges, de van néhány oldal ahol ezt a lehetőséget nem tartottam fontosnak(kérdések, állásajánlatok)
- Oldalsávban jelenjen meg az archívum és a legutóbbi bejegyzések, illetve címkefelhő: az oldalsávhoz még hozzáadtam egy gombot amely a szerkesztés oldalra mutat.
- Lábléc: impresszum, hírlevélre való feliratkozás, kapcsolat, gyakran ismételt kérdések: Végül a hírlevelet nem tartottam fontosnak, illetve hiányzik a gyakran ismételt kérdések oldala, mivel az oldal használata egyértelmű, amennyibe bármilyen kérdés, illetve felmerülne a kapcsolatok oldalon küldhetnek a látogatók emailt.

3.1.2. Nem funkcionális követelmények

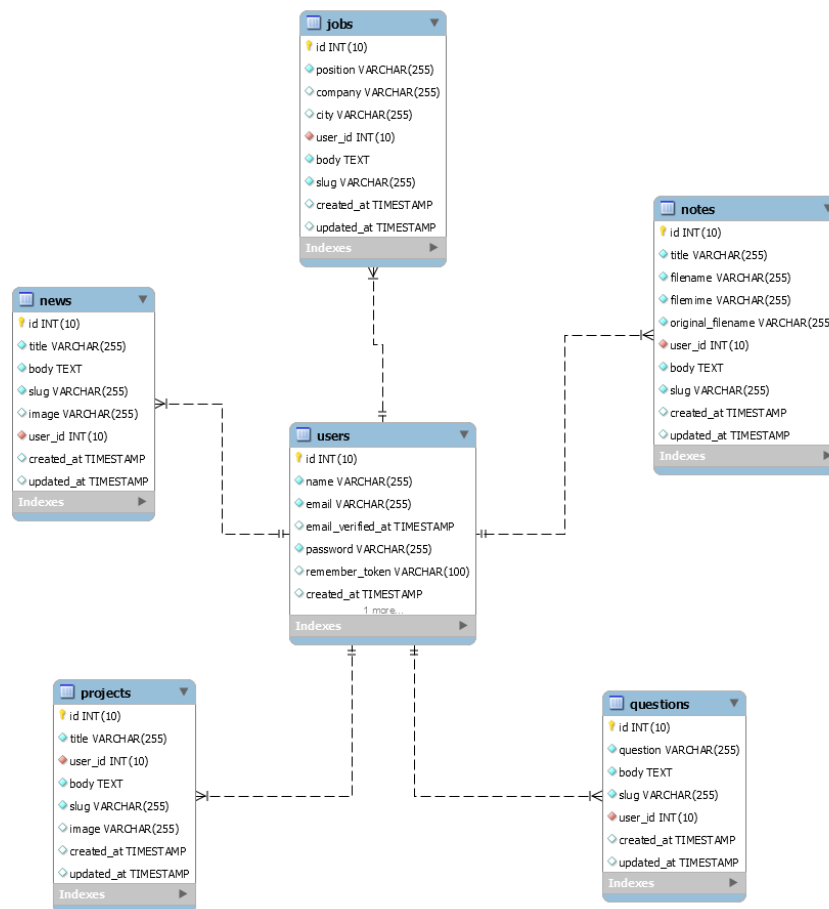
- Használhatóság: a weboldal mindenki számára könnyen használható legyen, a weboldal legyen akadálymentesített, felhasználói dokumentáció nélkül is könnyen olvasható legyen.
- Teljesítmény: a weboldal legalább 100 látogató kiszolgálására legyen egyidejűleg alkalmas, egy oldal betöltése a linkre való kattintás után legfeljebb 2 másodperc legyen.
- A weboldal legyen elérhető a hét legalább 95%-ban amennyiben karbantartás zajlik azt jelezzük a felhasználók számára: mivel még nem volt lehetőségem éles környezetben tesztelni az oldalt ezért ennél és a az előző pontnál a válasz még várat magára.
- Az oldal készítése során az MVC mintát kövessük: ezt a Laravel keretrendszer által könnyű megvalósítani, mivel az egész keretrendszer a Model-Nézet-Vezérlő architektúrára épül.
- A felhasználók jelszavai ne legyenek visszafejthetők: a Laravel felhasználók azonosítására egy saját komponenst használ, ezzel pedig a fejlesztés sokkal könnyebbé válik.
- A regisztráció során végezzünk ellenőrzést az email cím érvényességét illetően.
- A weboldal több eszközről is legyen elérhető, illetve azokon könnyen olvasható: a weboldal elkészítése során reszponzív sablonokat használtam és alakítottam át, hogy nem csak számítógépen de mobiltelefonon és tableten is jól használható legyen.
- Az alkalmazáshoz tartozó adatbázis a későbbiekben legyen bővíthető: az adatbázisba további táblákat hozhatóak létre a különböző kategóriáknak.

3.2. Adattárolás

A feladat során szükség volt a bejelentkezett felhasználók bejegyzéseiknek, fájljaiknak, hozzászólásaiknak való tárolására. Ennek a megvalósítására külön táblázatokat hoztam létre a jegyzetek, a projektek, az állások és a kérdések valamint egyéb bejegyzések számára, ezeket pedig összekapcsoltam a felhasználók táblájával.

A táblázatok kezelésére a MySQL Workbench programot használtam, amely grafikus felületen segíti az adatbázisunk adminisztratív feladatainak elvégzését.

A *user* táblához tehát 5 másik tábla is, amely az oldalakhoz tartozó adatokat tárolja. A *user* táblát a Laravel már eleve létrehozta, illetve emellett kapunk még egy táblát amely a jelszó visszaállításra szolgál és email címeket tárol.

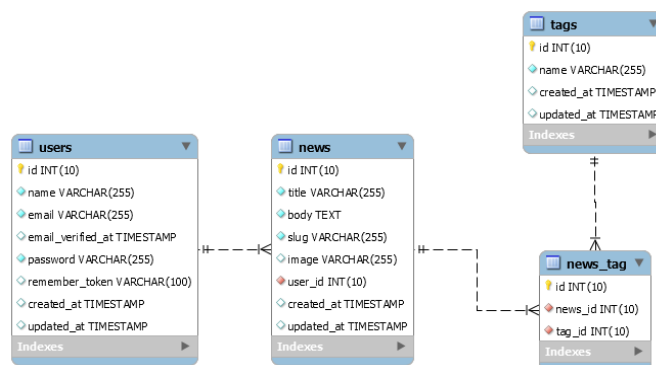


3.1. ábra. A felhasználók 5 kategóriában hozhatnak létre bejegyzéseket

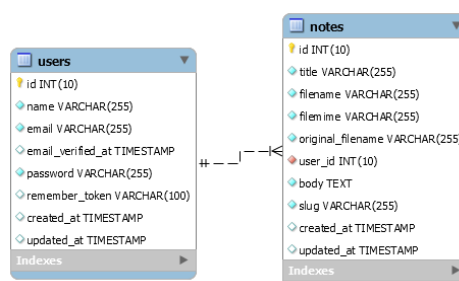
A regisztrált felhasználóknak lehetőségük van bejegyzéseket létrehozni 5 kategóriában. A *news* tábla tárolja a felhasználók által létrehozott hírek és események adatait. A *projects* tábla a projektek adatait. A *notes* tábla tárolja a hallgatók által feltöltött jegyzeteket, különböző formátumokban. A *questions* tábla tárolja az oldalon létrehozott kérdéseinket. Ezeket a táblákat azért bontottam szét hogy a későbbiekben tudjam bővíteni az oldalak funkcióit a felhasználói igények szerint. Ezen kívül van még egy *jobs* tábla, amely állásajánlatok adatait tárolja.

Az oldal a keresés és a csoportosítás megkönnyítésére címkéket használ, a tárolására

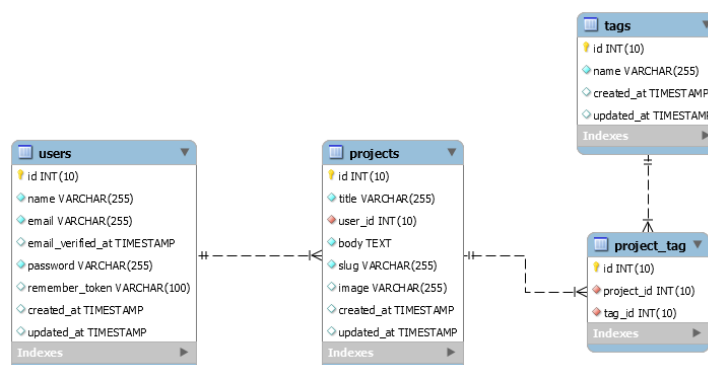
használt *tags* tábla közvetetten össze van kötve a kategóriák táblájával.



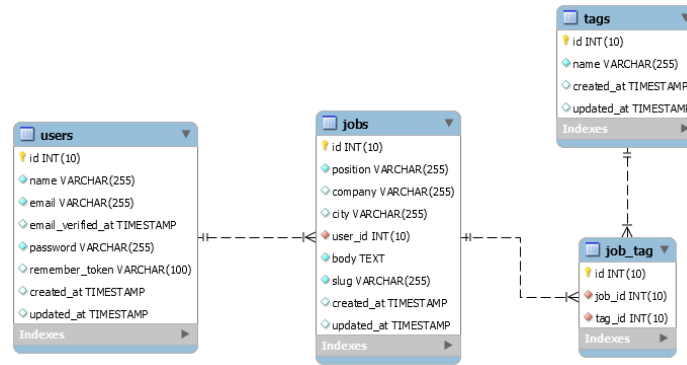
3.2. ábra. A *news* tábla az oldalon található hírek adatait tároja



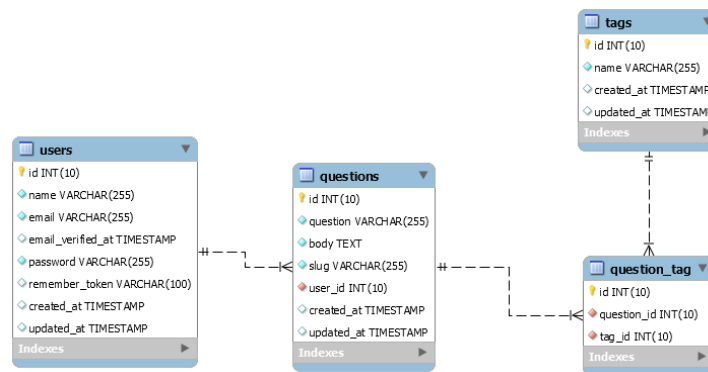
3.3. ábra. A *notes* tábla a hallgatók által feltöltött jegyzetek tárolására szolgál



3.4. ábra. A *projects* tábla, a diákok projekt ötleteit



3.5. ábra. A *jobs* tábla, az oldalon a hallgatók megoszthatnak egymással munkaaajánlatokat is



3.6. ábra. A *questions* tábla a fórumon elhangzott kérdések számára

3.2.1. Adatok szinkronizációja

A Laravel migration egy megoldás a táblák létrehozására és szinkronizálására a keretrendszeren belül. A migráció létrehozza helyettünk a tábláinkat ehhez csupán néhány beállításra van szükség. A `.env` konfigurációs fájlba be kell írunk az adatbázisunk adatit, amely ezután létrehozza a kapcsolatot a kiszolgálóval.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=homestead
DB_USERNAME=homestead
DB_PASSWORD=secret
```

3.7. ábra. Az adatbázis konfigurálása a `.env` fájlban

A `php artisan make:migration create_tablename` paranccsal létrehozhatunk migrációs fájlt, amely tartalmaz egy `up()` és egy `down()` függvényt. Az `up()` részen beállíthatjuk a táblánk oszlopait, a `down()` pedig eldobja a táblánkat.

```
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateFlightsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('flights', function (Blueprint $table) {
            $table->increments('id');
            $table->string('name');
            $table->string('airline');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::drop('flights');
    }
}
```

3.8. ábra. A parancs kiadása után ezt az eredményt kapjuk

3.2.2. Adatok feltöltése

Az adatok feltöltésére és manipulálására a Laravel Eloquent ORM komponensét használtam. Minden adatbázis tábla rendelkezik egy neki megfelelő "Modellel" amely az adott táblával való interakcióra szolgál. A model lehetővé teszi az adatok táblába való feltöltését, lekérdezést, kapcsolatokat jelöl a táblák között.

Ezt összekapcsolhatjuk az adatbázis migrációval, ha a `php artisan make:model ModelName` parancs mellé kiadunk egy `-m` vagy `--migration` kapcsolót:

```
php artisan make:model Flight --migration

php artisan make:model Flight -m
```

3.9. ábra. A kiadott parancs után ...

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

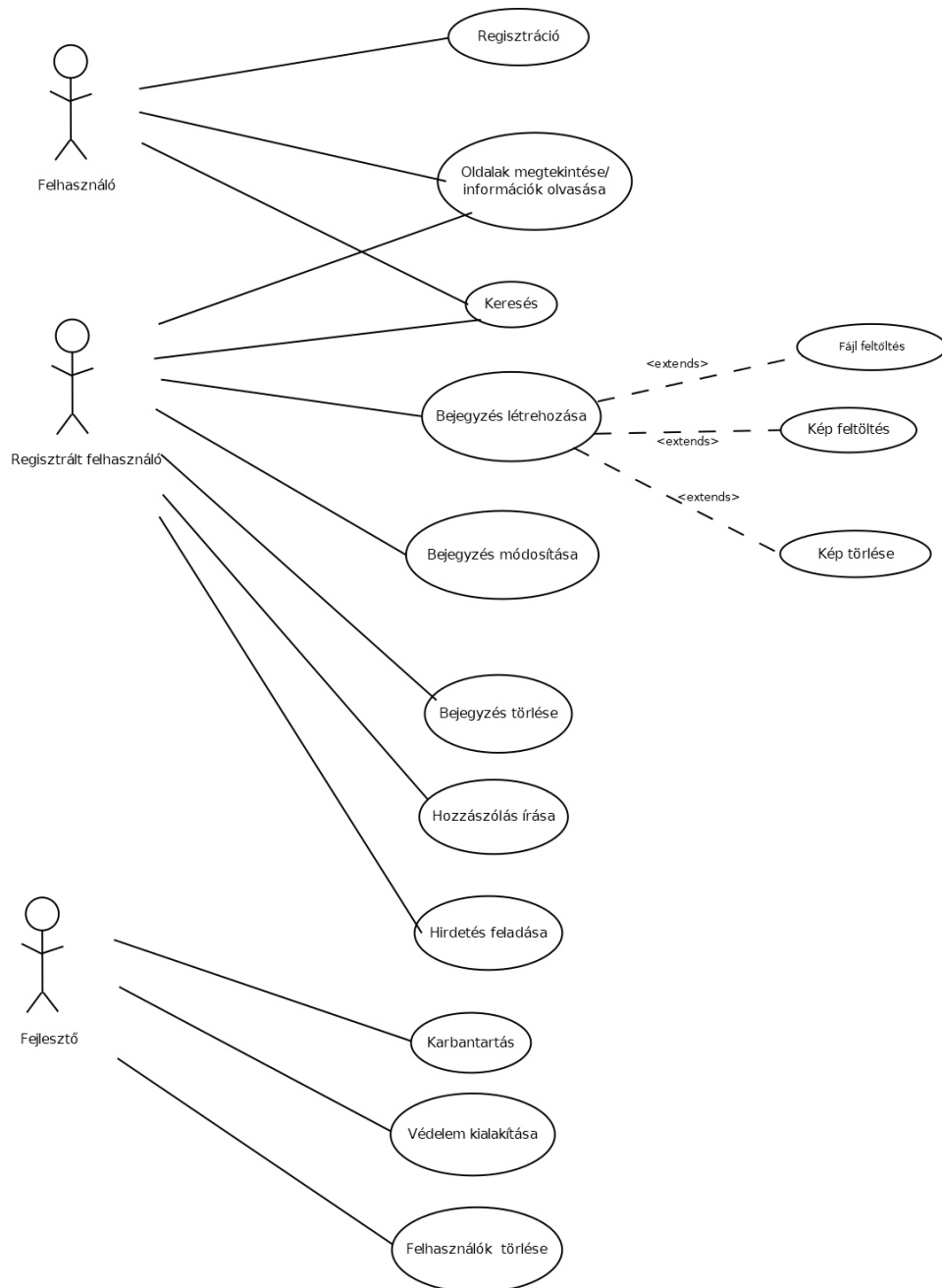
class Flight extends Model
{
    //
}
```

3.10. ábra. ... megkapjuk az objektumunkat

3.3. Felhasználói szerepkörök

A weboldalon alapvetően két felhasználói kör jelenik meg. Ezeket use case diagramon ábrázoltam.

- NEM REGISZTRÁLT FELHASZNÁLÓ: Az oldal bárki számára olvasható, azonban a nem regisztrált felhasználók nem tudnak hozzászólásokat írni, fájlokat feltölteni és bejegyzéseket készíteni, illetve ezeket módosítani.
- REGISZTRÁLT FELHASZNÁLÓ: Az oldal minden funkcióját el tudják érni. Szerkeszthetnek bejegyzéseket, tölthetnek fel fájlokat, hozzászólhatnak a bejegyzésekhez. Ezen kívül tudják a saját bejegyzéseiket módosítani, törölni. A biztonság növelését szem előtt tartva a bejegyzések alatti kommenteket ellenőrizhetik és törölhetik
- FEJLESZTŐ: Feladat az oldal megfelelő működésének biztosítása, a védelem kialakítása, a felhasználók adatainak ellenőrzése, felvitele és módosítása, valamint új alkalmazások létrehozása az oldalon belül.



3.11. ábra. A felhasználók use case diagramja

3.4. Grafikus felület

A grafikus felület fejlesztése során két sablont használtam, egyet a bejelentkezett felhasználók számára, és egyet az olvasható felülethez. Erre azért volt szükség mert szerettem volna ha bejegyzések szerkesztése egy sokkal letisztultabb oldalon valósul meg, de a mindenki számára elérhető felület sokkal látványosabb, ezáltal vonzza a látogatókat. A feladat készítése során Bootstrap keretrendszert használtam, amely előre megírt multifunkcionális eszközöket kínál, ami által gyorsabbá és könnyebbé válhat a munka. A HTML struktúra és CSS tulajdonságok mellett számos JavaScript bővítményt is tartalmaz melyek igen rugalmasak.

Az oldalak kialakításánál fontos a reszponzivitás, hogy minden eszközön olvasható legyen az oldal, ebben is segítségünkre van a Bootstrap. Szerencsére ma rengeteg előre megszerkesztett

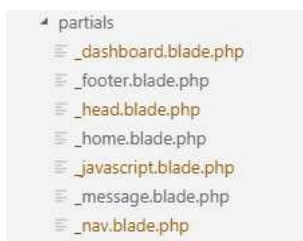
template-et találunk az interneten, én is ezeket használtam. A kész felületet részletesen a használatra vonatkozó fejezetben mutatom be.

3.4.1. Oldalak kialakítása Laravel használatával

Az oldalak kialakításánál megkönnyíti feladatunkat, hogy a Laravel beépített sablonmotorját az ún. Blade-et használjuk. Használatának előnye a sablon öröklődése, amivel szakaszokat alakíthatunk ki. Mivel az oldalak legtöbb esetben ugyanazt az általános elrendezést használják ezért érdemes .blade.php kiterjesztésű fájlokat használnunk, aminek lényege hogy az oldal nézetét egyszerű PHP kódban állítják össze és tárolják azokat, ami megkönnyíti a betöltést az oldalaknál.

A feladatomban például külön részre bontottam a fejléc és lábléc kialakítását, így ezeket elég volt csak egyszer létrehozni és úgy betölteni oldalra ezek kívül az oldalak tartalmaznak még egy borítóképet is. Egy példa a programból:

A különböző részleteket a view/partials mappában hozzuk létre. ebben az esetben 7 kis részre voltak bontva az oldalak: oldalsáv, lábléc, a fejléc, a borítókép, a javascriptek számára létrehozott rész, az oldal üzeneteit visszaadó rész, illetve a navigációs menüt tartalmazó részlet.



A mappában található elemeket pedig beimportáljuk egy másik .blade.php kiterjesztésű fájlba. A példa a main.blade.php-t mutatja be.

```
<!doctype html>
<html lang="{{ app()->getLocale() }}">
    <head>
        //Beinkludáljuk a <head> -ben megjelenő alapbeállításokat
        @include('partials._head')
        //Ide kerülnek majd az oldalakon használt stílusok @section
        ('stylesheets') @endsection utasítások közé
        @yield('stylesheets')
    </head>

    <body>

        <div class="super-container">
            @include('partials._nav') //a navigációs menü helye

            @include('partials._home') //borítókép
            @include('partials._message') //válasz üzenetek

            @yield('content') //a tartalom helye

            @include('partials._footer') //a lábléc helye
```

```

</div>
@include('partials._javascript')    //az oldalon használt
    javascript fájlok
@yield('scripts')    //beállítható scriptek

</body>

</html>

```

Jól látható hogyan hoztuk létre a sablonunkat apró részletekből. Az `@include` utasítással importáljuk a részleteket `@yield` utasítás pedig arra szolgál hogy betöltsön egy HTML tartalmat amelyet `@section` `@endsection` utasítások között adunk meg. Jelen esetben az oldal lényegi tartalmáról van szó. Természetesen több ilyen sablont is létrehozhatunk. Azt hogy az oldalunk milyen sablonhoz igazodjon az `@extends` utasítással adhatjuk meg pl.: `@extends('main')`.

3.4.2. Felhasználói élmény növelése az oldalon

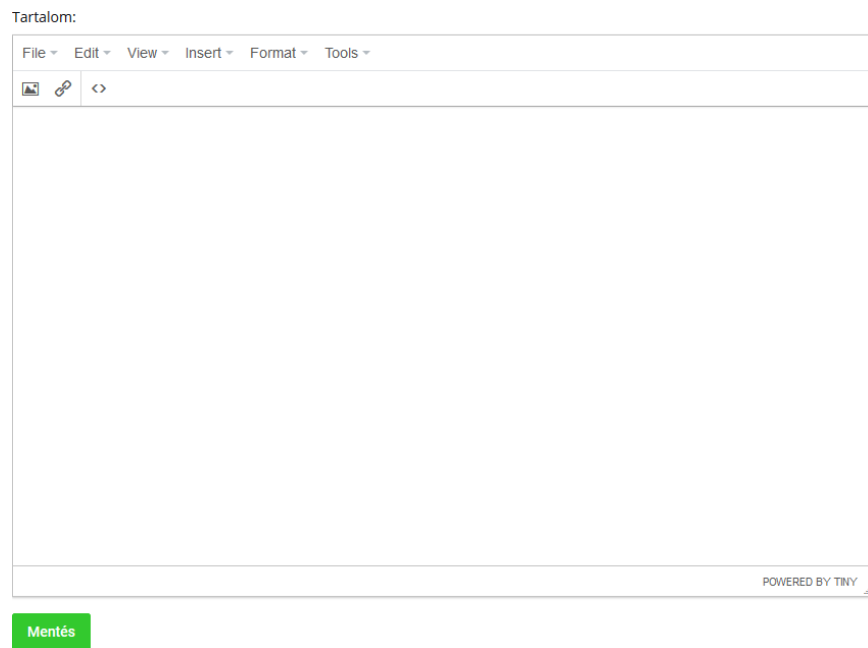
Az oldalon a bejelentkezett felhasználók szerkesztési felülete és nem a vendég felhasználók olvasói felülete külső megjelenésében jelentősen eltér. Ennek oka hogy a szerkesztő felülete letisztult és egyszerű, hogy könnyen használható és átlátható legyen. Az olvasó felület sokkal látványosabb annak érdekében hogy ezáltal is több látogatót vonzzon az oldal. Más hasonló témában készült oldalakon nem láttam hogy a külső megjelenésre ekkorra energia lett volna fordítva, én úgy gondoltam hogy a tartalom mellett ez is nagyon fontos, különösen hogy az oldal leginkább a 18-25 év közötti korosztályt célozza meg.

Az oldal címkéket is használ, ezek kiválasztásához pedig egy jQuery alapú selectbox komponenst a Select2-t választottam. Ez felülírja a HTML select boxát ezáltal sokkal könnyebben válogathatunk a felkínált opciókból



3.12. ábra. A címkék kiválasztása

A weboldal kapcsán természetesen felmerül az igény hogy felhasználók kedvükre szerkeszthessék az általuk létrehozott bejegyzéseiket. Erre több úgynevezett WYSIWYG szerkesztőt találunk, az egyik ilyen alkalmazás a TinyMCE amelyet könnyedén beállíthatunk a böngészőnkben. segítségével a bejegyzés készítője beszúrhat linket, fájlokat, képeket az oldalára, mint bármelyik szövegszerkesztőben. Ez különösen fontos volt az oldal esetében, mivel kódok és linkek beszúrására is szükség van, ezt pedig kényelmessé kell tenni a felhasználók számára.



3.13. ábra. A szerkesztő felülete

4. fejezet

Módszertan

4.1. Laravel keretrendszer

Bár a korábbi fejezetekben már megemlítettem, itt szeretném megragadni az lehetőséget hogy bővebben írjak az általam használt keretrendszerről, valamint arról hogyan használtam egyes segédprogramjait az alkalmazásomban. A Laravel egy ingyenes, nyílt forráskódú PHP keretrendszere, amelyet Taylor Otwell hozott létre, és alapvetőleg a modell-nézet-vezérlő architektúrát követi. A Laravel több olyan segédprogramot használ amely lehetővé teszi számunkra, az adatbázisunk kezelését, a weboldalunk gyorsabb betöltődését és az üzenetküldő rendszerünk kialakítását.

4.1.1. Rövid története

A Laravel első béta verziója 2011. június 9-én jelent meg, majd nem sokkal később még ugyanabban a hónapban megjelent a Laravel1. Ebből a verzióból még hiányoztak a kontrollerek ez megakadályozta, hogy valódi MVC (modell-nézet-vezérlő) alapú keretrendszer legyen.

Pár hónappal később ezt javították és megjelent a Laravel 2, amelyben megtalálható volt már a Blade nevű sablonrendszer, ami mint korábban említettem megkönnyíti a munkát az oldalak megjelenésének kialakítását.

A Laravel 3-at 2012. februárjában adták ki egy sor új funkcióval, beleértve az artisan nevű parancssori interfészt, illetve támogatást nyújtott több adatbázis kezelő rendszerhez, az adatbázis-migráció segítségével. Ebben a verzióban már megtalálható volt egy csomagkezelő rendszer a Bundles, amely lehetővé tette hogy a fejlesztők megoszthassák egymással az általuk készített kódjaikat.

A Laravel 4 vagyis Laravel Illuminate, 2013 májusában jelent meg. Ez a Laravel teljes körű átírása, a composer által elosztott külön csomagokba, amely alkalmazásszintű csomagkezelőként szolgál. Megjelent a Database Seeding, amely segíti az adatbázis adatokkal való feltöltését. Megjelent a Mail támogatása is, ami hozzájárul alkalmazásunk különböző rendszerekkel való összekapcsolásához. Az üzenetsorok (Queues) által lehetőség nyílt az időigényes kérések későbbi időpontra való halasztásához.

A Laravel5 a Laravel ma is használt változata 2015. ben jelent meg. A Scheduler által lehetőségünk van rendszeres időközönként végrehajtott feladatok ütemezésére, a FlySytem nevű csomag segítségével lehetőségünk van távoli adattárolásra. A Notification által lehetőségünk van az email küldés mellett az SMS és Slack üzenetek küldésére is. A Socialite csomag lehetőséget biztosít az OAuth autentikáció használatára.

4.1.2. Blade

A Blade a Laravel által nyújtott egyszerű, nagy teljesítményű sablonmotor. Az összes Blade-nézet valójában egyszerű szöveges formában készül. A feladat során külön sablonokat készítettem a weboldalam egyes részeihez, mint a fejléc és a lábléc valamint a minden oldalon használt JavaScript fájlt is egy külön részbe tettem a view/partials fájlban találhatóak. Ezen kívül az oldalak típusaihoz külön sablont készítettem, amiben elhelyeztem különböző részeket, és alkalmaztam az oldalakra annak megfelelően hogy az admin felülethez vagy az olvasható információs felülethez tartozik, azon belül is egy index oldal vagy a bejegyzés részére készült oldal. Ezzel lényegesen meggyorsítottam a munkámat. Mivel az előző fejezetben hoztam már rá példát ezért itt nem teszem meg.

4.1.3. Migrations

Az adatbázis migráció, lehetővé teszi az adatbázis sémájának könnyed módosítását, például egy csapatmunka esetében is. Több adatbázis kezelővel jól működik, és szorosan kapcsolódik a Lara Schema Builderéhez. Az adatbázisomat a migráció segítségével hoztam létre illetve a módosítottam a dolgozat során.

Példa a programból, a projekteknek szánt táblázatra:

```
public function up()
{
    //Tábla létrehozása
    Schema::create('projects', function (Blueprint $table) {
        $table->increments('id');
        $table->string('title');
        $table->integer('user_id')->unsigned();
        //tábla oszlopainak meghatározása
        $table->text('body');
        $table->timestamps();
    });

    Schema::table('projects', function ($table) {
        $table->foreign('user_id')->references('id')->on('
            users')->onDelete('cascade'); //idegen kulcs ö
            szekapcsolás a felhasználók táblájával
    });
}

public function down()
{
    Schema::dropIfExists('projects');
    Schema::dropForeign(['user_id']);
}
```

Ezek után a `php artisan:migrate` paranccsal létrehozunk a táblát az adatbázisban. A táblázathoz ugyanígy hozzáadhatunk egy mezőt is.

```
public function up()
{
    Schema::table('projects', function (Blueprint $table) {
        $table ->string('image')->nullable()->after('slug');
    });
}
```

4.1.4. Controller

A kontrollerek jellemzően az eseményeket és a felhasználói műveleteket dolgozza fel és válaszol rájuk, a modellben történő változásokat is kiválthat. Ennek kérelmezési logikáját a Laravel egyetlen osztályba csoportosítja. A Laravel egyik előnye, hogy CRUD (Create – létrehozás, Read – olvasás, Update – frissítés, Delete –törlés) útvonalakat rendel az alkalmazásunkhoz csupán egyetlen paranccsal.

Kiadva a `php artisan make:controller ProjectsController -resource` parancsot, a következőt kapjuk.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

//Létrejött a kontrollér
class ProjectsController extends Controller
{

    public function index()
    {
        //az adatok kilistázása
    }

    public function create()
    {
        //új rekord létrehozása szükséges űrlap létrehozása
    }

    public function store(Request $request )
    {
        //az új adat mentése a táblázatban
    }

    public function show($id )
    {
        //az adat megjelenítése
    }
}
```

```

public function edit($id )
{
    //az adat módosításához szükséges űrlap létrehozása
}

public function update(Request $request , $id )
{
    //az adat módosítása a táblában
}

public function destroy($id )
{
    //adat törlése a táblából
}
}

```

Jól látható a CRUD struktúra, az egyes metódusoknál.

4.1.5. Route

Lehetővé teszi hogy az alkalmazásnak küldött kérések a nekik megfelelő válasszal térhessenek vissza. Az útvonalakban megadhatunk különböző paramétereket, csoportokat definiálhatunk, elnevezhetjük azokat. Az útvonalainkat egy fájlban határozhatjuk meg a web.php-ban.

Az alábbi példában arra látunk példát, hogyan hozhatunk létre POST, GET, és DELETE kéréseket, valamint a resource() függvény minden kéréshez ad egy útvonalat. Az útvonalainkat nevesíthetjük is a name() függvénnyel, amivel könnyebben tudunk hivatkozni az útvonalunkban, például egy form kitöltésénél. Az Auth egy segédosztály amely segít létrehozni a felhasználó autentikációjához szükséges útvonalakat.

```

Route::group(['middleware'=>['web']], function(){

    Route::get('hirek', ['uses' => 'PostController@getNewsIndex', 'as' => 'post.newsindex']);
    Route::post('/comment/store', 'CommentsController@store')->name('comment.add');
    Route::delete('comment/{id}', ['uses' => 'CommentsController@destroy', 'as' => 'comment.destroy']);
    Route::get('comment/{id}/delete', ['uses' => 'CommentsController@delete', 'as' => 'comments.delete']);

    Route::resource('tags', 'TagController', ['except'=>['create']]);

    Route::resource('projects', 'ProjectsController');

});

Auth::routes();

```

4.1.6. Session

A Laravelben lehetőségünk van arra hogy a következő kérésre átvigyünk előzőleg bekért adatokat. Ezt a gyakorlatban úgy kell elképzelni hogy például kitöltünk egy HTML űrlapot, a megadott adatokat eltároljuk és amennyiben sikeres volt a küldés a következő felületen üzenetet kapunk vissza. Ezt a feladat készítése során a bejövő adatok tesztelésénél tudtam hasznosítani, azáltal hogy sikeres küldések esetére üzeneteket töltöttem be.

```
use Session;

public function destroy( $id )
{
    $project = Project::find( $id );

    $project -> delete();

    //Üzenet küldése a felhasználónak, a bejegyzés sikeres törlésé
    nek esetén
    Session::flash('succes', 'A bejegyzés sikeresen törölve.');
```

return redirect()->route('projects.index');

4.1.7. Pagination

Más keretrendszerekben a lapozás nehézkes lehet. A Larvel lapozási rendszere egységet a lekérdezővezérlővel és Eloquent ORM-mel ezáltal az adatbázis adatainak kényelmes, könnyen használható lapozását biztosítja. A létrehozott lapozó kompatibilis a Bootstrap keretrendszerrel. Ahhoz hogy az oldalunkhoz tudjuk alakítani a megjelenést publikussá kell tennünk az erre használt alapbeállításokban lévő fájlokat.

A projektek főoldalán 5 bejegyzés kerül egy oldalra, ezt következőképpen állíthatjuk be.

```
public function getProjectsIndex() {

    $projects = Project::paginate(5);
    return view('post.projectsindex')->withProjects( $projects
    );
}
```

És az oldalon így hivatkozunk rá.

```
<!-- Page Nav -->

{!! $projects -> links() !!}
```

4.1.8. Query Builder

A Laravel adatbázis lekérdezés készítője kényelmes felületet biztosít a lekérdezések létrehozásához és működtetéséhez minden támogatott adatbázis kezelő rendszeren. PDO paramétereket használ az alkalmazás, az SQL injekciós támadásokkal szembeni kiküszöbölésre.

```
public function getIndex(){

    //A főoldalonsökkenő beérkezése szerint csökkenő sorrendben
    jeleníti meg az adatokat
    //és egy limitet ad nekik
    $news = News::orderBy('created_at','desc')->limit(4)->get();
    $notes = Note::orderBy('created_at','desc')->limit(3)->get();
    return view('pages.welcome')->withNews($news )->withNotes($
        notes );
}
```

4.1.9. Eloquent ORM

Ez egy model implementáció adatbázisokhoz. Minden adatbázisnak van egy Model-je amely kölcsönhatásban áll a migrációs adatbázisunkkal. Ez teszi lehetővé az adatok táblázatokban történő lekérdezését és az új rekordok beillesztését.

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Project extends Model
{
    //A táblák egymás közötti kapcsolatát is itt állítjuk be
    public function tags(){

        return $this -> belongsToMany('App\Tag');

    }

    public function user(){
        return $this -> belongsTo('App\User');
    }
}
```

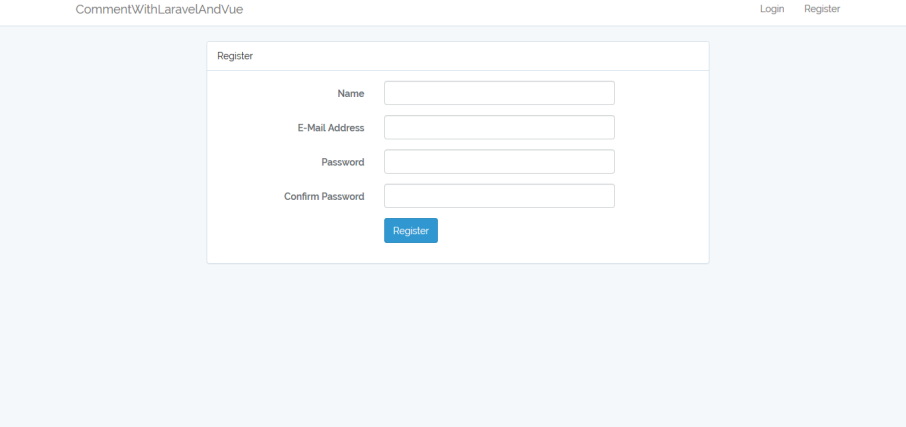
4.1.10. Authentication

A Laravel autentikációja megkönnyíti a felhasználói jogosultságok meghatározását. Egy parancsot kiadva létrehoz két felhasználói szerepkört, a vendég felhasználót és az azonosított felhasználót.

A jogosultság beállítása a következőppen történik:

```
class ProjectsController extends Controller
{
    public function __construct() {
        $this->middleware('auth');
    }
}
```

Innentől kezdve a projektek szerkesztése csak a felhasználó regisztrálók számára lehetséges. Ezzel a lehetőséggel létrejön több funkció is, a Laravel megoldja a regisztráció, a bejelentkezés és a jelszó visszaállítás problémáját, ezekhez létrehozza a megfelelő oldalakat amelyeket saját magunk átalakíthatunk. Az alapbeállítás és az egyedi oldal közötti különbség:

The image shows a web browser window with a light blue header. On the left of the header is the text 'CommentWithLaravelAndVue', and on the right are the links 'Login' and 'Register'. The main content area has a light blue background. In the center, there is a white rectangular box with a thin blue border. Inside this box, at the top, is the title 'Register'. Below the title are four input fields, each with a label to its left: 'Name', 'E-Mail Address', 'Password', and 'Confirm Password'. At the bottom of the box, centered, is a blue button with the word 'Register' in white text.

4.1. ábra. A Laravel alapbeállítása regisztrációs oldalon

The screenshot shows a web page for 'MEduLine' with a dark background. At the top right, there are links for 'Bejelentkezés' and 'Regisztráció'. The main content is a white modal box titled 'Bejelentkezés'. Inside the modal, there are two input fields: 'E-Mail Cím' (Email) and 'Jelszó' (Password). Below the password field is a checkbox labeled 'Jegyezzen meg' (Remember me). At the bottom of the modal is a large orange button labeled 'Bejelentkezés' and a link 'Elfelejtetted az email címed?' (Forgot your email?).

4.2. ábra. Az általam elkészített regisztrációs oldal

The screenshot shows a web page for 'CommentWithLaravelAndVue' with a light blue background. At the top right, there are links for 'Login' and 'Register'. The main content is a white modal box titled 'Login'. Inside the modal, there are two input fields: 'E-Mail Address' and 'Password'. Below the password field is a checkbox labeled 'Remember Me'. At the bottom of the modal is a blue button labeled 'Login' and a link 'Forgot Your Password?'.

4.3. ábra. Az eredeti bejelentkezés

4.4. ábra. És az oldalra készített

4.1.11. Mail

A Laravel egy tiszta egyszerű API-t biztosít az emailek küldésére a SwiftMailer könyvtár használatához, amely meggyorsítja az emailek küldését és fogadását helyi vagy felhőalapú szolgáltatásokon keresztül.

A példában a kapcsolatok oldal üzenetküldését mutatom be:

```
public function postContact(Request $request ){

    //A beérkező adatok ellenőrzés
    $this->validate ( $request ,[
        'email' => 'required|email',
        'subject' => 'min:3',
        'message' => 'min:10'
    ]);

    //A beérkező adatok eltárolása
    $data = array(
        'email' => $request ->email,
        'subject' => $request ->subject,
        'bodyMessage' => $request ->message
    );

    //Az eredmények elküldése az adott címre a Mail API függvény
    segítségével
    Mail::send('emails.contact', $data , function( $message ) use (
        $data ){
        $message->from( $data ['email']);
        $message->to('admin@gmail.com');
        $message->subject( $data ['subject']);
    });
}
```

```

    Session::flash('success','Az email el lett küldve');
    return redirect('/');
}

```

Az üzenetküldés ellenőrzéséről és beállításáról a következő fejezetben lesz szó.

4.1.12. HTMLPurifier

Ez egy hagyományos HTML szűrőkönyvtár amelyet PHP-ben írtak. Védelmet nyújt a HTML az XSS támadásokkal szemben. Ezt WYSIWYG szerkesztők ellenőrzésére használtam, mivel ennek ellenőrizetlen használatával könnyen bekerülhetnek veszélyes kódok az oldalra, amik használhatatlanná teszik azt.

A composer require mews/purifier parancsot kiadva telepítjük az alkalmazást. Ezután a config/app.php fájlba bemásoljuk a következőket:

```

'providers' => [
    //hivatkozás a kiszolgálóra
    Mews\Purifier\PurifierServiceProvider::class,
]

'aliases' => [
    //alias név beállítása
    'Purifier' => Mews\Purifier\Facades\Purifier::class,
]

```

Bállítjuk a szövegmezőkre.

```

$request->body = Purifier::clean( $request->body);

```

4.1.13. File Storage

A Laravel egy kiváló fájlrendszert hozott létre, mely könnyedén használható helyi tárolórendszerekhez, illetve az Amazon S3 és a Rackspace Cloud Storage online tárolórendszerekhez. A jegyzetek tárolása a következőképpen történik:

```

//Fájl feltöltése
public function store(Request $request )
{
    $note = new Note;
    $file = $request->file('filefield'); //ide kerül a file
    $extension = $file->getClientOriginalExtension(); //fájl
        kiterjesztésének vizsgálata
    Storage::disk('local')->put( $file->getFilename().'.'.$extension , File::get( $file )); //fájl elhelyezése a tá
        rhelyen
}

```

```

    $note -> filemime = $file -> getClientMimeType();    //sokszoros
        ítjuk
    $note -> original_filename = $file -> getClientOriginalName();
        //a fájl eredeti neve
    $note -> filename = $file -> getFilename().'.'.$extension ;    //
        a létrehozott szimbolikus link
}

//Letöltés
public function get_file( $filename ){

    $note = Note::where('filename', '=', $filename )->firstOrFail()
        ; //fájlnév kikeresése az adatbázisból
    $file = Storage::disk('local')->get($note ->filename);    //fájl
        lekérése a tárhelyről

    return (new Response( $file , 200))->header('Content-Type', $
        note -> mime); //a fájl egy példányának letöltése
}

```

4.1.14. Intervention Image

Egy PHP forrású képkezelő és manipuláló könyvtár. Ez egyszerűbb módot biztosít, képek létrehozásához, szerkesztéséhez, ezzel segítve hogy képeinket a weboldalunkhoz szabjuk ezáltal jobb minőségű képeket jelenítsünk meg az oldalunkon.

```

//Az oldalon megjelenő képek átméretezése
Image::make( $image )->resize(730,384)->save( $location );

```

4.2. Biztonság és védelem

4.2.1. Felhasználói adatok tárolása

A weboldal elkészítése során szükség volt arra, hogy a felhasználók adatait biztonságosan tárolhassam ezért a felhasználók jelszavait az adatbázisban sem jeleníthettem meg. Ez ugyebár a keretrendszer mellett adott volt, a jelszavak tárolására bcrypt hasht használ, tehát egy olyan függvényt amely tetszőleges hosszúságú üzenetből rögzített 128 bit hosszúságú üzenetet generál. Alapelve hogy valamennyi bemeneti bit összefügg valamennyi kimeneti bittel. Az ujjlenyomat kiszámítása előtt az üzenet kiegészíti, hogy annak hossza 512-vel osztva 448-at adjon, majd ezt kiegészíti az üzenetek hosszának 64 biten tárolt reprezentációjával. Ezt követően 512 bitenként feldolgozva képződik a 128 bites ujjlenyomat.

4.2.2. Jogosultságok beállítása

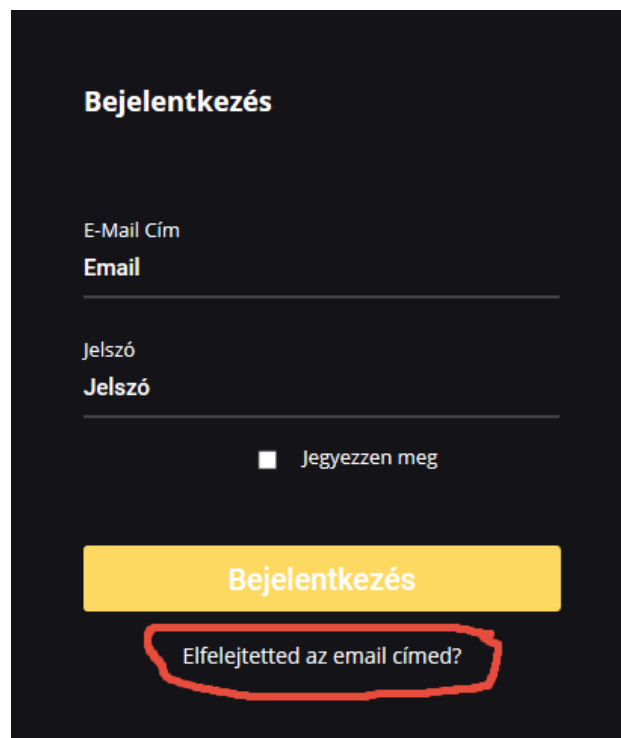
Az oldalon a felhasználók két csoportját különítjük el. Erre azért volt szükség mert az oldal ugyan szorosan kapcsolódik az egyetemhez, de emellett magához az informatika és a programozás témájához is így bárki regisztrálhat az oldalon, de ezen kívül a nem regisztrált felhasználók is olvashatják a weboldal bejegyzéseit illetve a jegyzetek letöltése számukra is

lehetséges, hiszen más egyetemek hallgatói is megfordulhatnak az oldalon. Ezen kívül pedig közösségi oldalalakon is látok arra példát, hogy a vállalkozásokhoz egyetemi csoportokban keresnek gyakornokokat. Az oldal tervezésének elején szó volt arról is hogy egyetemi email címhez kötöm a belépés lehetőségét a biztonság növelésének érdekében, ezt azonban idővel elvetettem.

4.2.3. Jelszó visszaállítás

A regisztrált felhasználóknak lehetőségük a van a jelszó visszaállításának kérésére amennyiben elfelejtették azt vagy nem tudnak belépni. Ehhez egy email címet kérek be, amely aztán kiküldi az megadott címre a visszaállításhoz szükséges oldalt, ahol egy form kitöltése során új jelszót állíthat be magának a felhasználó.

Az jelszó visszaállításának folyamata:



The image shows a dark-themed login form titled "Bejelentkezés". It contains two input fields: "E-Mail Cím" with the label "Email" and "Jelszó" with the label "Jelszó". Below the password field is a checkbox labeled "Jegyezzen meg". A large yellow button labeled "Bejelentkezés" is positioned below the checkbox. At the bottom of the form, a link "Elfelejtetted az email címed?" is highlighted with a red hand-drawn circle.

4.5. ábra. A bejelentkezés oldalán a linkre kattintunk

Jelszó visszaállítás

E-Mail

Email küldése

4.6. ábra. Megadjuk az email címünket

Jelszó visszaállítás

We have e-mailed your password reset link!

E-Mail

Email küldése

4.7. ábra. A jelszó elküldésre kerül

Hello!

You are receiving this email because we received a password reset request for your account.

Reset Password

If you did not request a password reset, no further action is required.

Regards,
Laravel

4.8. ábra. A visszaigazoló emailben kapunk egy linket...

Jelszó visszaállítás

E-Mail

E-Mail

Jelszó

Jelszó

Jelszó megerősítés

Jelszó

Visszaállítás

4.9. ábra. ...ami a jelszó visszaállításának oldalára mutat

4.2.4. Hozzászólások ellenőrzése

A weboldalon hozzászólások írhatók a bejegyzésekhez (ez a funkció részlegesen megoldott, a későbbi fejlesztésekről már egy korábbi részben írtam) annak érdekében hogy a hallgatók és a további felhasználók interakcióba léphessenek egymással, visszajelzéseket kapjanak egymástól. Ebben az esetben felmerülhet annak lehetősége hogy egy felhasználó nem megfelelő szavakat használ, esetleg nem minősíthető viselkedésével zavarja társait. Mivel az admin nem nézi rendszeresen az oldat ezért szükség van arra hogy ezek a megjegyzések időben lekerülhessenek az oldalról. Ezért a felhasználók a szerkeszthető felületen a bejegyzések alatt látják a beérkezett megjegyzéseket, és törölhetik azokat.

A hozzászólások törlése:

2 hozzászólás

Tartalom

Maecenas ullamcorper arcu a nulla suscipit venenatis. Nulla rutrum, sem et vulputate gravida, sapien justo pellentesque dui, vitae rhoncus mi lacus a velit.

Töröl

Nulla pretium risus scelerisque lorem cursus semper. Aliquam eleifend felis vel massa consectetur accumsan. Ut eu molestie nisl, ut scelerisque quam. Integer eu finibus odio. Fusce ullamcorper consequat porta.

Töröl

4.10. ábra. Ha a törlés gombra kattintunk...

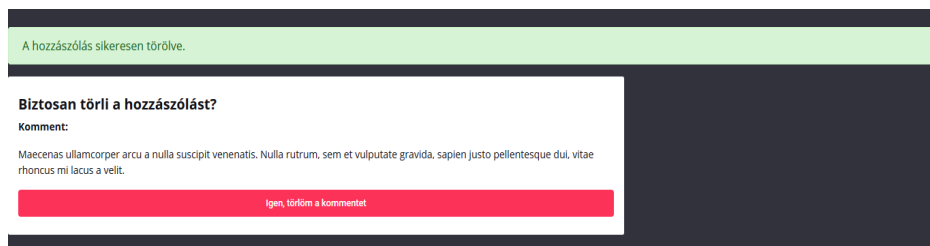
Biztosan törli a hozzászólást?

Komment:

Maecenas ullamcorper arcu a nulla suscipit venenatis. Nulla rutrum, sem et vulputate gravida, sapien justo pellentesque dui, vitae rhoncus mi lacus a velit.

Igen, törölöm a kommentet

4.11. ábra. ...megjelenik egy oldal, ahol megerősítjük törlési szándékunkat.



4.12. ábra. Visszajelzést kapunk a sikeres törlésről.

1 hozzászólás

Tartalom

Nulla pretium risus scelerisque lorem cursus semper. Aliquam eleifend felis vel massa consectetur accumsan. Ut eu molestie nisl, ut scelerisque quam. Integer eu finibus odio. Fusce ullamcorper consequat porta.

Töröl

4.13. ábra. Ezután a bejegyzés oldalán látjuk hogy a hozzászólás törölve lett.

4.2.5. WYSIWYG szerkesztő ellenőrzése

A weboldalon minden felhasználó a bejegyzéseit saját maga által készíti el, hozza létre, ehhez egy WYSIWYG editor áll rendelkezésükre ami úgy néz ki mint egy átlagos szövegszerkesztő. Ennek ellenőrzését meg kell oldanunk mivel általa bekerülhetnek az oldalunkra olyan HTML kódok amelyek kár okozhatnak benne. Ez az XSS támadási technikához kapcsolódik. A támadás a kliens és nem a szerver ellen irányul, és maga a weblap ami hordozza az XSS támadásra alkalmas kódot a támadásnak nem a célpontja, hanem annak eszköze. Ha az oldalunk ez ellen védtelen azt ki fogják használni. Ezzel az technikával ellophatják a felhasználók adatait, de az oldal működését is megakadályozhatják. Ennek kiküszöbölésére a HTML Purifier szűrőt használtam, amelyet az előző fejezetben már ismertettem.

4.2.6. Űrlapok kitöltése

Az űrlapok megfelelő ellenőrzésére azért van szükség mert a weboldalon előfordulhatnak bizonyos CSRF támadások. Ez a módszer azt használja ki, hogy az adott oldal megbízik a felhasználóban és nem ellenőrzi hogy adott lekérést valóban a felhasználó vitte-e véghez, mégpedig valóban a saját oldaláról. A CSRF támadások általában fórumokról indulnak, ahol az egyes hozzászólások szövegét az XSS-nél említett htmlentities-el már hatástalanították, így JavaScript kódot nem tudnak posztolni, ellenben képet igen. Mivel a Laravelben FormBuildert használunk ezért ez megoldott.

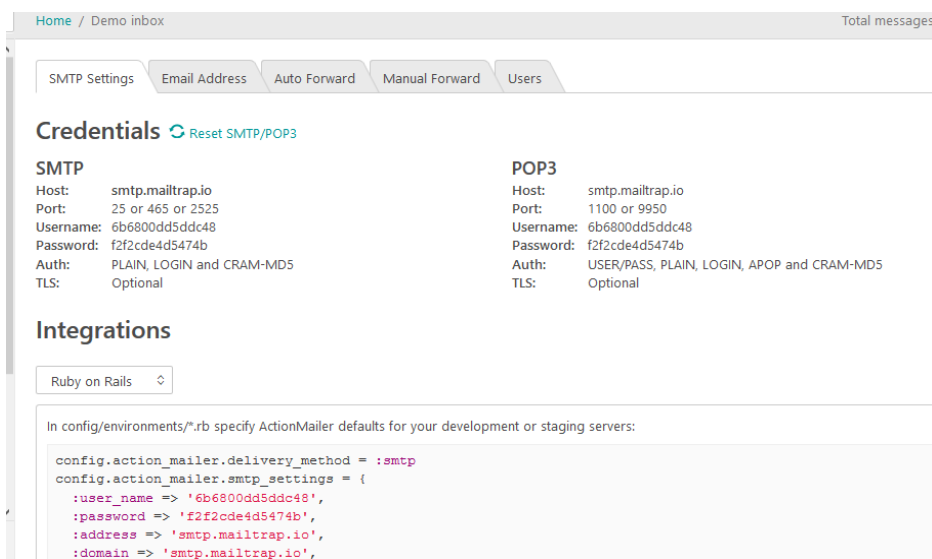
5. fejezet

Tesztelés

5.1. Üzenetküldés tesztelése

A weboldalon két esetben tudunk üzenetet küldeni a jelszó visszaállításának kérésekor, illetve a kapcsolatok oldalon küldhetünk üzenet a weboldal készítőjének.

Mielőtt használatba helyeznénk a weboldalt szükség van az üzenetküldés tesztelésére. Ehhez a folyamathoz segítségünkre van a Mailtrap, amely lényegében egy szimbolikus SMTP (Simple Mail Transfer Protocol) szerver. Segítségével ellenőrizhetjük az email üzeneteink tartalmának helyességét még mielőtt használatba helyeznénk az email küldésének lehetőségét az oldalon. Ahhoz hogy beállítsam az SMTP-t a Mailtrap által megadott felhasználó nevet és jelszót a .env mappába helyeztem, a titkosítást tls-re állítottam.

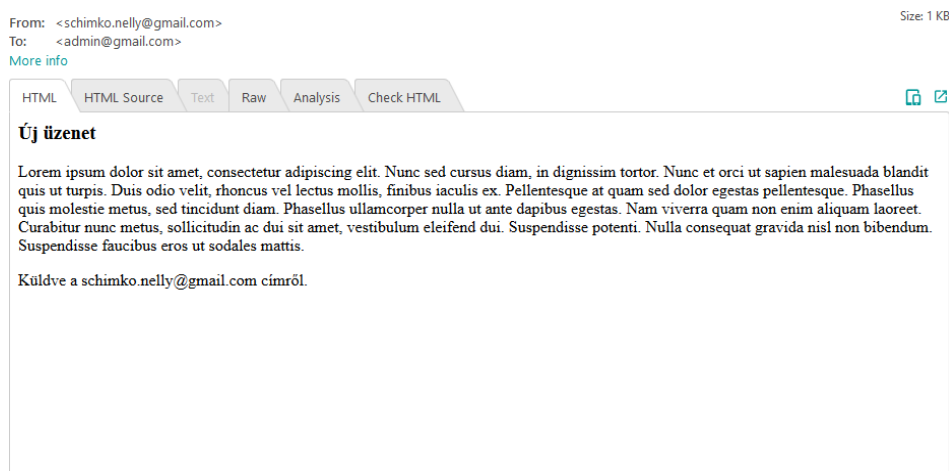


5.1. ábra. A kapott jelszót és felhasználónevet

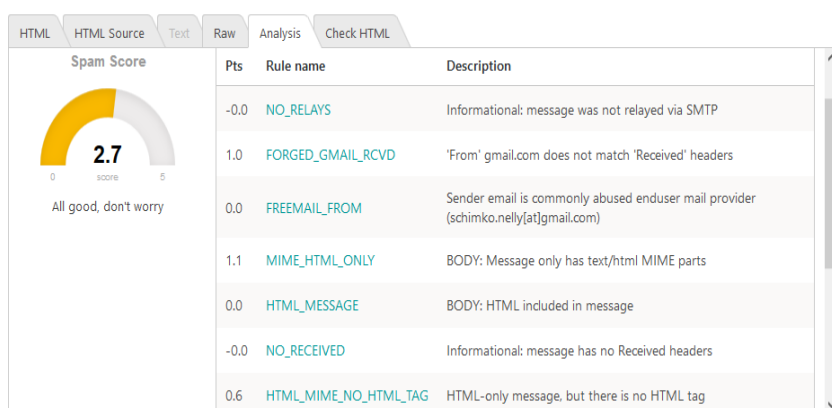
```
MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=6b6800dd5ddc48
MAIL_PASSWORD=f2f2cde4d5474b
MAIL_ENCRYPTION=null
```

5.2. ábra. megadjuk a .env fájlba

A beérkezett emailt lehetőségünk van HTML formában megtekinteni, továbbá információt kapunk arról is, hogy az üzenetünk tartalmaz-e vírust vagy szerepel-e feketelistán. Részletezést kapunk arról is, hogy mely HTML és CSS szabályok nem jelennek meg megfelelően a webes eszközeinken.



5.3. ábra. Egy kapcsolatfelvételi emailt látunk



5.4. ábra. Az emailünket tesztelhetjük mielőtt kiküldenénk

5.2. A készítő általi tesztelés

A tanulmányaim során nem volt lehetőségem programok tesztelésével foglalkozni, ezért az alkalmazást csak a gyakorlatban való kipróbálással tudtam tesztelni. Az oldalakat manuálisan feltöltöttem adatokkal azért, hogy ellenőrizni tudjam minden beállítás működik-e. Elsőként regisztráltam egy új felhasználónévvel az oldalra, majd minden kategóriában szerkesztettem néhány bejegyzést. Ellenőriztem hogy a publikus oldalon is működnek-e a beállítások, ha hibát találtam megpróbáltam kijavítani. A működésbeli problémákat szerencsére már nem kellett javítanom, a külső nézetben voltak hiányosságok és még vannak is, ezeket a közeljövőben szeretném kijavítani.

5.3. Felhasználó általi tesztelés

Annak érdekében, hogy az esetleg általam fel nem merült hibákat kijavítsam, az alkalmazást megmutattam egy tőlem sokkal tapasztaltabb szakembernek is. A tesztelése által több hibára, biztonsági részre is fény derült. Ezeket kijavítva készült el a jelenlegi állapotában az alkalmazás. Ezen kívül a lap külső megjelenésén is igazítottam, mint például a gombok színe, hogy a felhasználók a megfelelő funkciót társítsák hozzá. Ezen a hibák kezelését is teljessé tettem a hibaoldalakkal.

6. fejezet

Összefoglalás

6.1. Összefoglalás

A dolgozat az egyetem informatikus hallgatóinak szánt alkalmazást ír le, mely segítségükre lehet a tanulmányaik kezdetétől egészen a szakmai életben való elhelyezkedésükig. Az első fejezetben a meghatározott célcsoportokról, a programhoz hasonló kezdeményezésekről, illetve a keretrendszerrel szemben már megvalósult és még megvalósulatlan elvárásokról olvashatunk. Ezen kívül részletesen fel vannak tüntetve a program során használt technológiák programnyelvek.

A fejlesztői dokumentáció során részletezem, aza adatok tárolásának megvalósítását. Az adatbázis felépítését EER modell által mutatom be, illetve leírom hogyan valósul meg a táblák létrehozása és a amodelek összeköttetése a programban.

Egy alfejezetben részletes meghatározom a felhasználói szerepköröket, majd a következő fejezetben bemutatom a grafikus felületet.

A módszertanban bővebben ismeretetem az általam felhasznált laravel keretrendszert valamint a programtervezése során használt mvc szerkezeti mintát.

A biztonság és védelem részbem memutatom milyen biztonsági megoldásokat használ az oldal.

Egy további részt szentelek az általam és más felhasználó általi tesztelésnek.

A fejlesztés során számos tapasztalatra, ismeretre tettem szert, minden a webfejlesztés szempontjából, de önismereti szempontból is.

Rájöttem sokkal szervezettebbnek kell lennem és struktúráltabban dolgoznom, az alkalmazásokat érdemes részletesen megtervezni majd megvalósítani, mert ezzel is időd nyerhetünk, illetve érdemes jobba felmérni a képességeinket vagy hogy mennyi idő áll a rendelkezésünkre a projekt kapcsán.

A weboldal elkészítése során megismertem a php alapú Laravel keretrendszert és annak komponenseit, amely számomra eddig még új volt és azért választottam, mert a jövőben is szívesen foglalkoznék webfejlesztéssel, ezt a keretrendszer pedig népszerű elég gyakran használt vállalkozásoknál.

A téma választásának oka pedig az volt, hogy úgy éreztem hiánypotló lenne egy ilyen oldal az informatikus hallgatók számára. Ezen felül az interneten keresgélve úgy láttam elég ritka egy ilyen sok témakört magába foglaló, összetett oldal egyetemi hallgatók számára. A legtöbb ilyen kezdeményezés közösségi oldalakon valósul meg, de valószínűleg nem eléri minden hallgatót, ráadásul érdemesnek tartom ezt egy struktúráltabb formában megvalósítani.

6.2. Summary

This dissertation describes an application, targeting the computer science students of the university, which could help them from the start of their studies until they can start a job in the profession. In the first section you can read about the target groups, similar initiatives and the accomplished/to-be-accomplished goals towards the framework. Besides these the technologies and programming languages used during the development of the program are also noted.

Throughout the developer documentation I specify the storage of data. The structure of the database is presented through an EER model while I also describe the process of the creation and connection in the program.

In a subsection I detail the different actors for the use cases and in the next chapter I present the graphical interface.

In the methodology I describe Laravel framework and the MVC model in detail, which I used during the program designing.

In the security and protection section, I describe what security solutions are used on the site.

There is a section dedicated for the test results of other users as well as my own.

During the development, I gained lots of knowledge, including both web development and self-knowledge.

I realised that I need to be way more organised, and I have to work in a more structured form. I should make a detailed specification for the application before creating them, as this can help in gaining time. Also I need to better assess my skills and how much time do I have for a project.

During the development process. I got to know the php based Laravel framework and its components which were new to me. I decided to use them, because I would like to work with web development in the future too, and I know this framework is wildly popular among companies.

The reason for picking this topic was that I felt the need for a site like this among fellow programmer students. Besides this, searching on the Internet, I realised there is a lack of sites which contain this many topics for university students. Most initiatives like this are accomplished using social media sites, and while they probably not only not reach every student, I think using a better structured platform is a better idea.

Irodalomjegyzék

- [1] THE WORLD'S LARGEST WEB DEVELOPER SITE,
<https://www.w3schools.com/>
- [2] The PHP Framework For Web Artisans,
<https://laravel.com/>
- [3] Wikipedia-Laravel,
<https://en.wikipedia.org/wiki/Laravel>
- [4] Colorlib-Course,
<https://colorlib.com/wp/template/course/>
- [5] Bootstrap Dash - Stellar Admin Pro,
<https://www.bootstrapdash.com/product/stellar-admin-template/>
- [6] How to Build a Blog with Laravel,
<https://www.youtube.com/playlist?list=PLwAKR305CR0-Q90J---jXVzb0d4CDRbVx>
- [7] Parsley, the ultimate JavaScript form validation library,
<http://parsleyjs.org/>
- [8] TinyMCE - Build Beautiful Content for the Web with Tiny,
<https://www.tiny.cloud/>
- [9] Select2 - The jQuery replacement for select boxes,
<https://select2.org/>
- [10] mailtrap - Safe Email Testing for Development Teams,
<https://mailtrap.io/>
- [11] MeWebStudio - Muharrem ERİN - Purifier,
<https://github.com/mewebstudio/Purifier>
- [12] Intervention Image,
<http://image.intervention.io/>
- [13] PHP,
<http://php.net/>
- [14] Wikipédia - PHP,
<https://hu.wikipedia.org/wiki/PHP>
- [15] Bootstrap,
<https://getbootstrap.com/>
- [16] Wikipedia - Bootstrap,
[https://en.wikipedia.org/wiki/Bootstrap_\(front-end_framework\)](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework))

- [17] MySQL,
<https://www.mysql.com/>
- [18] Wikipédia - MySQL,
<https://hu.wikipedia.org/wiki/MySQL>
- [19] Wikipédia - HTML,
<https://hu.wikipedia.org/wiki/HTML>
- [20] Wikipédia - CSS,
https://hu.wikipedia.org/wiki/Cascading_Style_Sheets
- [21] Wikipédia - SQL,
<https://hu.wikipedia.org/wiki/SQL>
- [22] Wikipédia - jQuery,
<https://hu.wikipedia.org/wiki/JQuery>
- [23] Apache,
<https://httpd.apache.org/>
- [24] Programnyelvek Portál,
<http://nyelvek.inf.elte.hu/>
- [25] BME VIK Wiki,
<https://vik.wiki/Kezdőlap>
- [26] Stackoverflow,
<https://stackoverflow.com/>
- [27] Create a Comments System with Laravel And Vuejs,
<https://www.cloudways.com/blog/comment-system-laravel-vuejs/>

Adathordozó használati útmutató

A DVD két mappát tartalmaz a La_TEX mappa tárolja a szakdolgozat PDF fájlját, illetve az ahhoz tartozó fájlokat, képeket. A másik MEduLine mappában található maga az elkészült alkalmazáshoz tartozó összes fájl. Az oldal még nem érhető el az interneten, csak localhoston működik.

Ahhoz hogy telepítsük az alkalmazást a következő lépéseket kell megtennünk.

1. Győződjünk meg arról, hogy a gépünkön megtalálható-e a php és az Apache alkalmazás, illetve valamilyen rendelkezik-e a számítógép valamilyen adatbázis kezelővel. Ezeket állítsuk be.
2. Miután kész vagyunk menjünk fel a <https://getcomposer.org/download/> oldalra és töltsük le a composer PHP kezelő eszközt.
3. Ezután hozzunk létre a parancssorból egy új Laravel projectet: `composer create-project --prefer-dist laravel/laravel` projektnév
4. Miután ezzel is megvagyunk másoljuk a mappában található fájlokat.
5. A `.env` fájlban állítsuk be az üzenetküldéshez és az adatbázis eléréshez szükséges adatokat:

Miután végeztünk a beállítással nézzük meg a `/config` mappában található `app.php` fájlunka és ellenőrizzük hogy megtalálhatóak-e benne a következők:

```
'providers' => [  
    Collective\Html\HtmlServiceProvider::class,  
    Intervention\Image\ImageServiceProvider::class,  
    Mews\Purifier\PurifierServiceProvider::class,  
],  
  
'aliases' => [  
    'Image' => Intervention\Image\Facades\Image::class,  
    'Form' => Collective\Html\FormFacade::class,  
    'Html' => Collective\Html\HtmlFacade::class,  
    'Purifier' => Mews\Purifier\Facades\Purifier::class,  
]
```

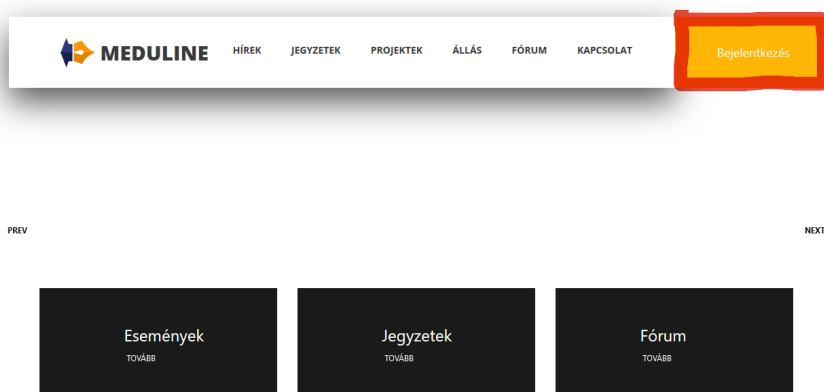
Ha ezt nem találjuk akkor egészítsük ki a fájlt.

Ezután adjuk ki a következő parancsokat:

1. `php artisan migrate`
2. `composer install`

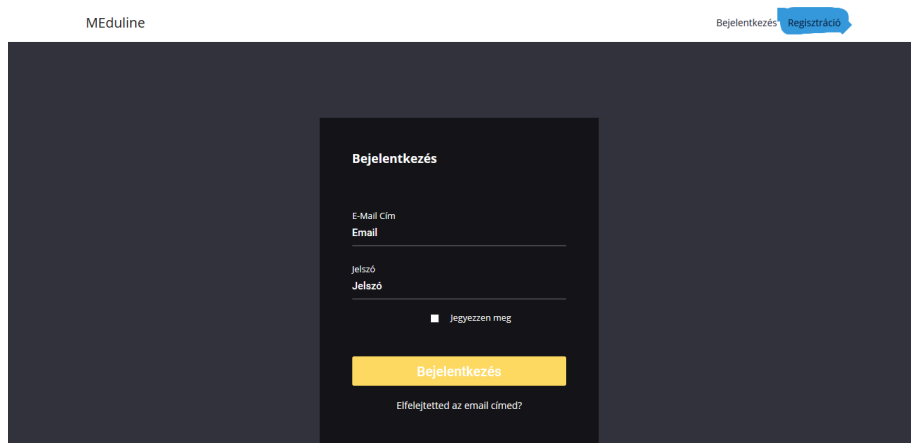
6.2. Summary

Amikor felmegyünk a localhostra a főoldal jön be:

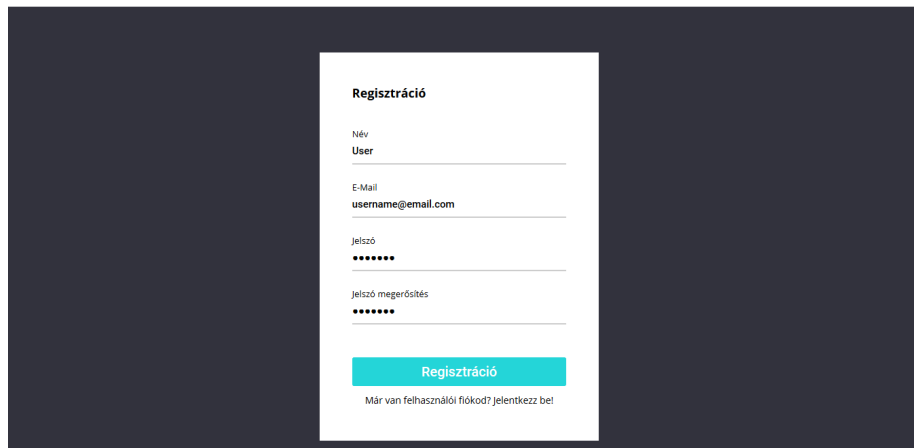


6.1. ábra. A kezdő oldalunk

A bejelentkezés linkre kattintva bejön az a belépés oldal:



6.2. ábra. Lépünk be vagy regisztráljunk



Regisztráció

Név
User

E-Mail
username@email.com

Jelszó

Jelszó megerősítés

Regisztráció

Már van felhasználói fiókod? Jelentkezz be!

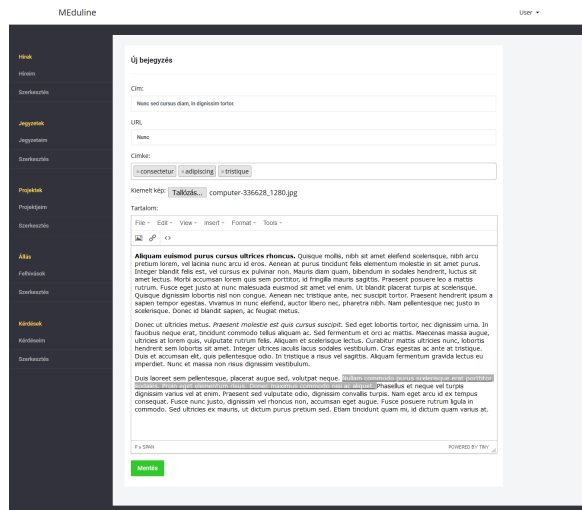
6.3. ábra. Regisztráció az oldalon

Jobb oldalt fent a regisztráció gombra kattintva regisztrálhatunk, regisztráció utána pedig az adminisztrációs felületre lépünk. Itt baloldalt találunk egy menüt a különböző témakörök-höz. Minden témához tartozik egy oldal amin szerkeszthetjük bejegyzéseinket, és egy oldal ami felsorolja az általunk kiírt bejegyzéseket. A bejegyzéseket egyenként megnézhetjük, módosíthatjuk őket. A szerkesztet bejegyzések megjelennek az oldalon.

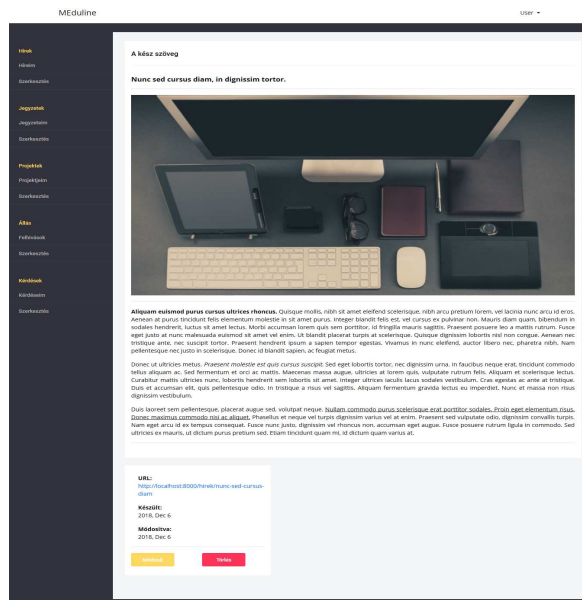


6.4. ábra. A regisztráció és belépés után megjelenő adminisztrációs felület

6.2. Summary



6.5. ábra. A bejegyzés szerkesztése



6.6. ábra. A létrejött bejegyzés adatai



6.7. ábra. Az oldalon megjelenő bejegyzés