Szegedi Szent Benedek School of Business Technikum

Minden Egy Helyen

Szakképesítés neve: Szoftverfejlesztő -és tesztelő

Témavezető: Babinszky Mónika, Bodrogi Péter

Készítők: Trájer Balázs, Lipták Dominik, Rácz Norbert

Tartalom

[Laravel 4](#_Toc160780710)

[Mi az a Laravel? 4](#_Toc160780711)

[Laravel projekt kezdése 4](#_Toc160780712)

[Mi az a Composer? 6](#_Toc160780713)

[Backend 7](#_Toc160780714)

[Új adatbázis csatlakozás 7](#_Toc160780715)

[MVC – Model View Controller 8](#_Toc160780716)

[Model (Modell) 8](#_Toc160780717)

[View (Nézet) 9](#_Toc160780718)

[Controller (Vezérlő) 9](#_Toc160780719)

[Hitelesítés 9](#_Toc160780720)

[Regisztáció logika 9](#_Toc160780721)

[Bejelentkezés logika 10](#_Toc160780722)

[Kijelentkezés logika 10](#_Toc160780723)

[Admin felület 10](#_Toc160780724)

[Profilok kezelése 11](#_Toc160780725)

[Helyszínek kezelése 12](#_Toc160780726)

[Hozzáférési jogok 12](#_Toc160780727)

[Profil felület 12](#_Toc160780728)

[Profil adatai - Profilkép 13](#_Toc160780729)

[Profil adatai – Név és cégszám 13](#_Toc160780730)

[Profil adatai – Jelszó 14](#_Toc160780731)

[Helyszínek kezelése 14](#_Toc160780732)

[Helyszín törlése 15](#_Toc160780733)

[Frontend 16](#_Toc160780734)

[Regisztráció nézet 16](#_Toc160780735)

[Bejelentkezés nézet 17](#_Toc160780736)

[Admin oldal 18](#_Toc160780737)

[Profil oldal 18](#_Toc160780738)

[Források 18](#_Toc160780739)

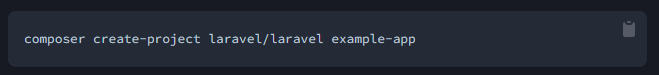
# Laravel

## Mi az a Laravel?

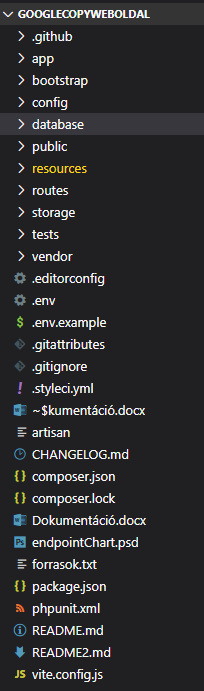
A Laravel egy nyílt forráskódú, PHP programozási nyelven írt webes alkalmazásfejlesztési keretrendszer. Az alapjaiban olyan eszközök és struktúrák gyűjteménye, amelyek segítenek fejlesztőknek a webes alkalmazások gyors és hatékony építésében. A Laravel célja, hogy egyszerűsítse a fejlesztési folyamatot, lehetővé téve a kód újra felhasználhatóságát és a könnyű karbantarthatóságot. A Laravel számoskiegészítőt, csomagot és eszközt kínál, amelyek lehetővé teszik a fejlesztők számára a gyors és hatékony fejlesztést. Ez a keretrendszer népszerűvé vált a fejlesztők körében a könnyű használhatósága, a kiterjedt dokumentációja és a nagy közössége miatt.

## Laravel projekt kezdése

Egy Laravel projekt úgy kezdődik, hogy elsősorban egy kiegészítő programot be kell szerezni, ez a program a Composer. A Composer -ről későbbiekben lesz szó. A Laravel -t legegyszerűbben a Laravel weboldalán megtalálható dokumentáció alapján lehet elkezdeni. Ha le van töltve a Composer, akkor egy tetszőleges helyen kell nyitni egy parancssort vagy Git Bash -t, majd az alábbi kódot kell bemásolni vagy beírni.

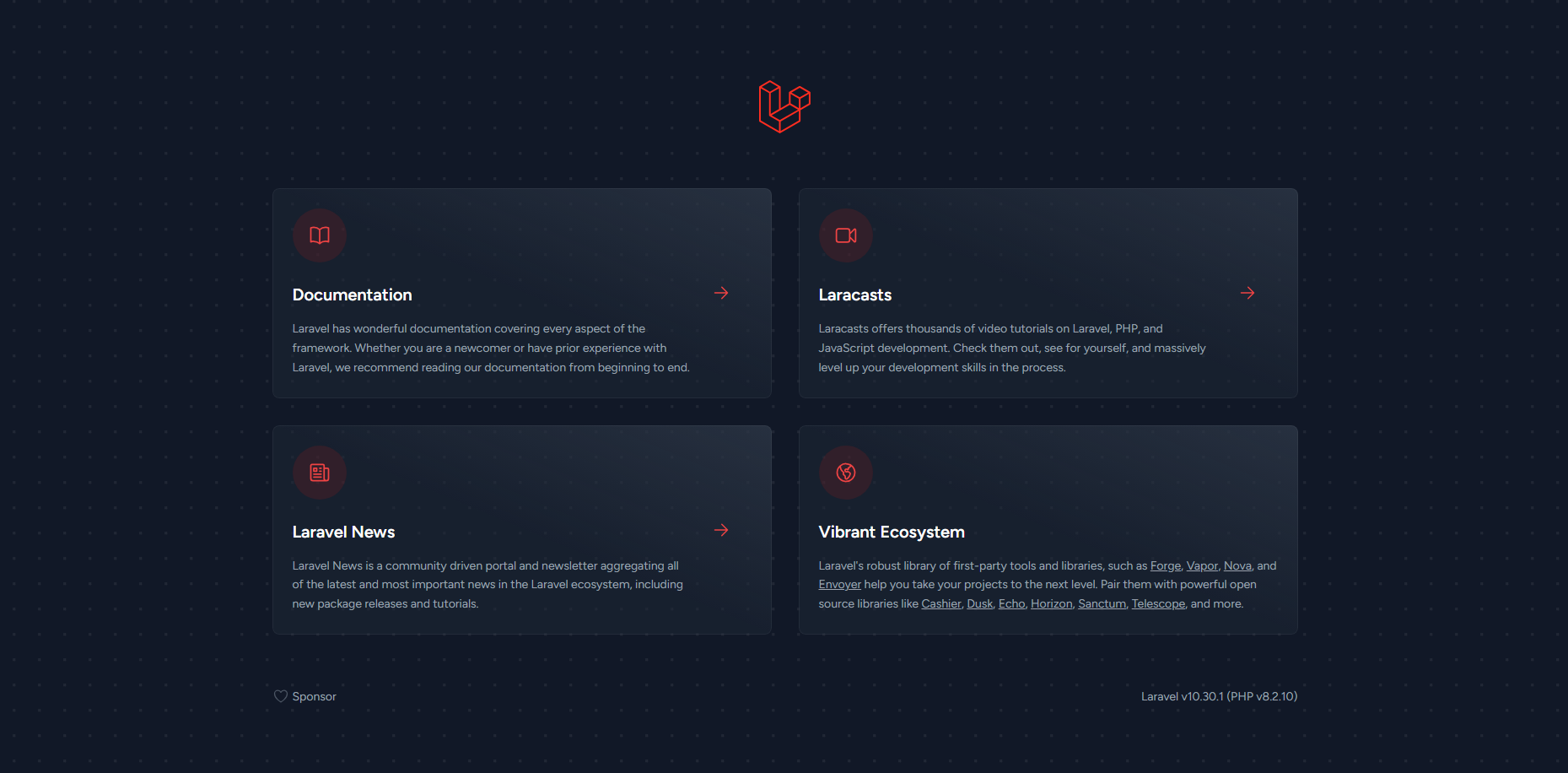


Hogyha ezt a kódsort futtatjuk, akkor a jelenleg megnyitott mappába egy „*example-app*” nevezetű projektet fog létrehozni, ez a folyamat számítógéptől függően változó ideig tarthat.



Ha a projekt létre lett hozva, akkor valahogy így fog kinézni a projekt mappanézete, leszámítva pár filet, amit utólag lett hozzáadva. A Laravel tartalmaz egy php szervert is, ezt a „*php artisan serve*” parancsot a terminálba kell beírni és így lehet elindítani.

Hogyha el van indítva a szerver, akkor ez a weboldal fog megjelenni:



Ez az előre legenerált főoldal, amin pár Laraveles weboldal található meg és lehet támogatni a Laravel kitalálóját, Taylor Otwell -t.

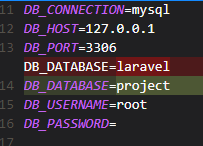
## Mi az a Composer?

A Composer egy népszerű függőségkezelő eszköz a PHP programozási nyelvhez. A fő célja, hogy segítsen a PHP-alapú projektjeinkben az egyes könyvtárak és csomagok kezelésében, valamint ezek függőségeinek hatékony kezelésében. Ez az eszköz lehetővé teszi a fejlesztők számára, hogy egyszerűen telepítsék, frissítsék, és kezeljék a különböző külső könyvtárakat, csomagokat, és azok verzióit a projektjükben. A Composer egy JSON-alapú konfigurációs fájlt használ, amely tartalmazza a projekt függőségeit, valamint azok verzióit és kapcsolódó beállításokat. Az eszköz automatizálja a külső függőségek letöltését az internetről és azok integrálását a projektbe. Például a Composer segít PHP keretrendszerek, könyvtárak vagy egyedi csomagok telepítésében a Laravel, Symfony, vagy más PHP-alapú projektek számára. Emellett gondoskodik a verziók kezeléséről és az összes függőség feloldásáról, hozzáogy biztosítsa a megfelelő működést a projektben. A Composer jelentős mértékben megkönnyíti a fejlesztők számára az alkalmazások fejlesztését és karbantartását, valamint lehetővé teszi a könnyű és hatékony együttműködést az nyílt forráskódú PHP projektek között.

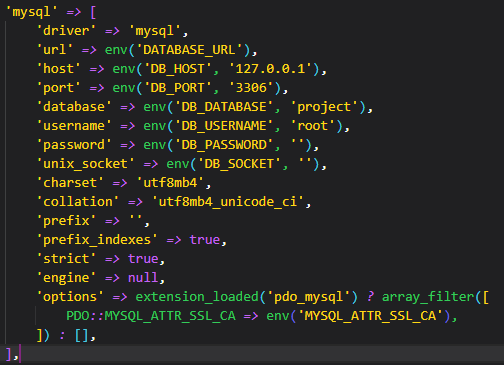
# Backend

## Új adatbázis csatlakozás

Mivel a Laravel keretrendszer hozzá lett adva a projekthez, ezért új csatlakozást kell létrehozni.



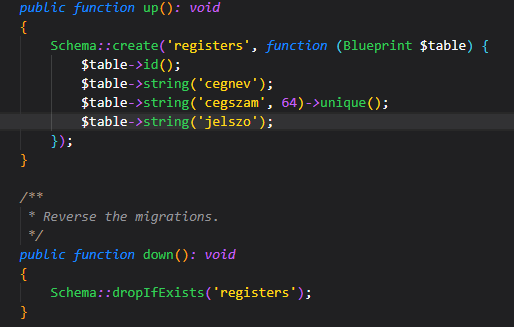
Ahhoz, hogy hogy hozzácsatoljuk, az „*.env*” fileban át kell írni a „*DB\_DATABASE*” értékét az alap „*laravel*” -ről a mi adatbázis nevünkre, ami ebben az esetben „*project*”. Emellett még a „*config/database.php*” fileban is át kell írni az adatbázis nevét és felhasználó nevét a helyesre.



Miután ezek a konfigurációs fileok át lettek írva, egy migrációt kell létrehozni, ezt a következő paranccsal lehet:



Ez létrehoz egy nyers migrációs filet, ebben meg kell adni a tábla oszlopok neveit és egyéb tulajdonságait.



Ha be lett állítva a migrációs file, akkor a következő paranccsal lehet létrehozni a táblát:



Ez a rész hibára tud futni, mivel a php tele van kiegészítőkkel, amit manuálisan kell aktiválni, ebben az esetben a „*extension=pdo*” -t kell aktiválni.

## MVC – Model View Controller

Az MVC egy tervezési minta, amelyet a szoftvertervezésben alkalmaznak a kód strukturálására és szervezésére. Az MVC elkülöníti egy alkalmazás három fő részét, hogy könnyebbé tegye a fejlesztést és karbantartást. Ezek a három fő rész:

### Model (Modell)

A modell felelős az alkalmazás üzleti logikájáért és adatmanipulációjáért. Itt találhatók az adatstruktúrák, adatbázis-interakciók és más olyan komponensek, amelyek közvetlenül kapcsolódnak az alkalmazás adattárolásához és feldolgozásához. A modellek kezelik az adatok integritását és logikáját.

### View (Nézet)

A nézet felelős az adatok megjelenítéséért és az interfész kialakításáért. Ez a rész teszi lehetővé, hogy az alkalmazás a felhasználóval kommunikáljon. A nézetek olyan elemeket tartalmaznak, mint az HTML-sablonok, amelyekkel az adatokat megjelenítik a felhasználó számára.

### Controller (Vezérlő)

A vezérlő kezeli a felhasználói bemeneteket és szabályozza a kommunikációt a modell és a nézet között. Amikor a felhasználó valamilyen műveletet hajt végre (például gombnyomás vagy űrlap beküldése), a vezérlő kezeli az eseményt, módosítja a modellt (szükség esetén) és frissíti a nézetet.

## Hitelesítés

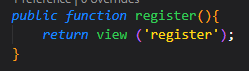
Ezen a weboldalon a cégeknek kell csak regisztrálniuk, abban az esetben, ha posztolni akarnak. A Laravelben létezik az „*php artisan make:auth*” parancs, ami előre létrehozza a hitelesítéseket a megadott modellre, de a projekt esetébe nem volt használva.

### Regisztáció logika

Ahhoz, hogy a regisztrációhoz hozzáférjen a felhasználó, létre kell hozni egy controllert. Ezt az alábbi kóddal lehet létrehozni:



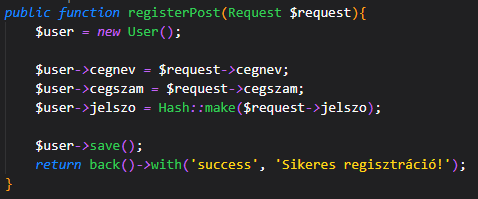
Ebben a fileban eleinte csak egy osztály található.



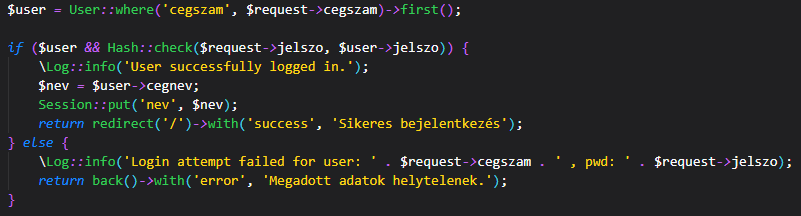
A fenti funkcióval el kell látni, mert ezt fogja meghívni amikor át akarunk menni a „*/register*” oldalra. Ahhoz, hogy működjön, a „*/routes/web.php*” -ban meg kell hívni:



Hogyha megkapja a „post” -ot a szerver, akkor egy másik funkciót küld

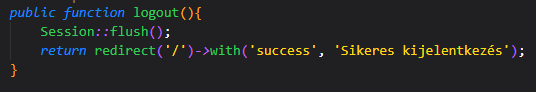


### Bejelentkezés logika



A bejelentkezési logika lényege, hogy a „*user*” model alapján létrehoz egy új usert, ez megkapja a megadott cégszámot, majd ellenőrzi, hogy a usernek megadott jelszó az megegyezik e a lekért jelszóval, ha megegyezik, akkor egy változónak átadja a nevet, amit majd késöbb a „*session*” fog használni, és visszairányítja a felhasználót a főoldalra.

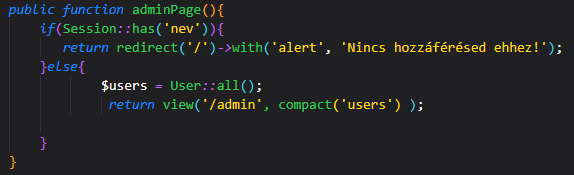
### Kijelentkezés logika



A kijelentkezés szimplán csak törli a törli a munkamenetet.

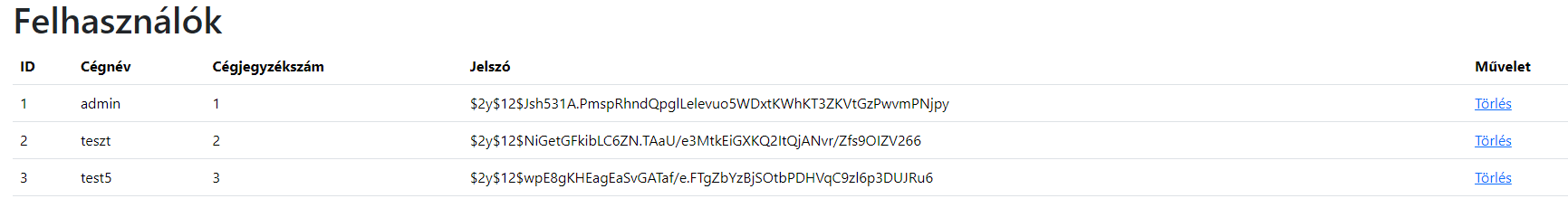
## Admin felület

Az admin felület létszükséges egy ilyen projekt során, ezért itt is implementálásra kerül.



### Profilok kezelése

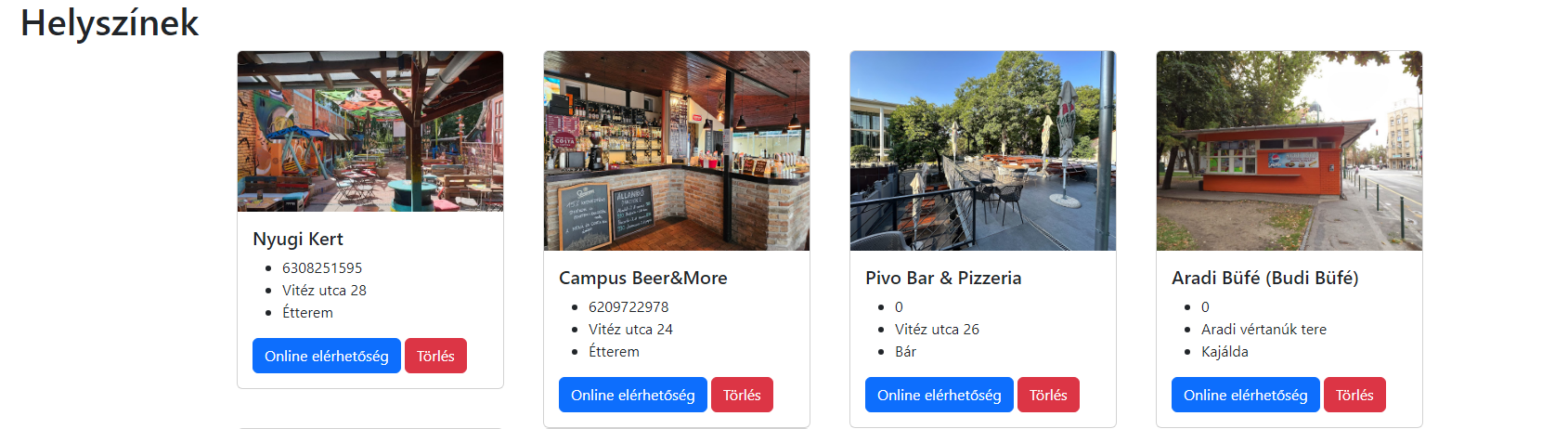
Első sorban minden regisztrált felhasználó kiírásra kerül, ehhez meg kell hívni a „*user*” modellt.



### Helyszínek kezelése



Helyszínek minden adattal együtt kilistázásra kerülnek.



## Hozzáférési jogok

Minden weboldalhoz, amihez az általános felhasználó (azaz a nem cégek és admin) nem férhet hozzá, ahhoz egy vizsgálatot kell rakni.



Itt azt vizsgálja a program, hogy a munkamenetnek van e „*nev*” vagy „*admin*” változója, ha van akkor visszavezeti a főoldalra a felhasználót.

## Profil felület

A profil felület bonyolultabb, mint az admin, mert itt csak a „*session*” -ben eltárolt felhasználó adatai jelennek meg és csak azokat lehet módosítani.

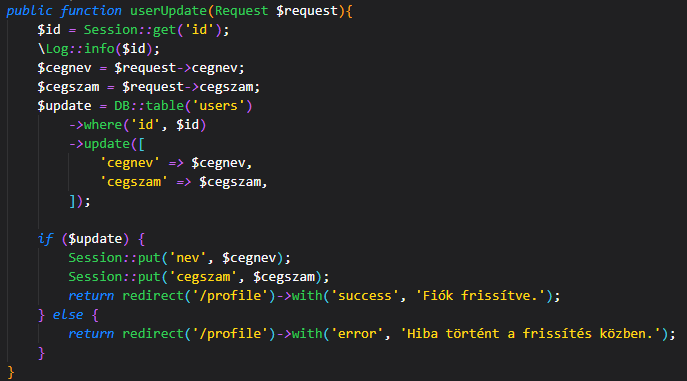
A profil felületet 3 részre lehet szétszedni.

### Profil adatai - Profilkép



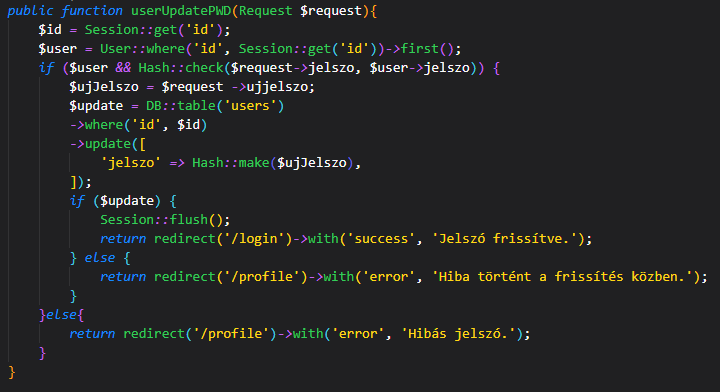
Az „*updatePic*” funkció egy formmból kéri át az értéket, majd feltölti az adatbázisba. Ha minden jól ment, akkor sikerrel fog visszajönni és a munkamenetnek átadja az új értéket.

### Profil adatai – Név és cégszám



A cégnevet és a cégszámot ugyan úgy lehet megváltoztatni, mint a profilképet, egy formmból kéri át az adatokat, majd az id alapján feltölti az adatbázisba, majd a munkamenetnek átadja az értékeket, hogy azonnal látszódjanak.

### Profil adatai – Jelszó



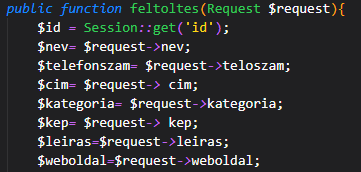
A jelszó nagyrészben ugyan úgy működik, mint a fentiek, csak ennél ellenőrzi, hogy a jelenlegi jelszó megegyezik e az adatbázisban hozzárendelt jelszóval, ha megegyezik akkor a formmban lévő új jelszót feltölti, majd a bejelentkezési oldalra visz tovább.

## Helyszínek kezelése

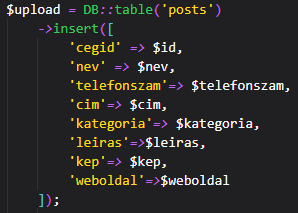
A weboldal lényege, hogy helyszíneket töltenek fel a vállalkozások, így ez elengedhetetlen. A helyszíneket formon keresztül lehet feltölteni. Egy helyszínnek 7 tulajdonságát kell megadni, a 2 id -t automatikusan megkapja.

****

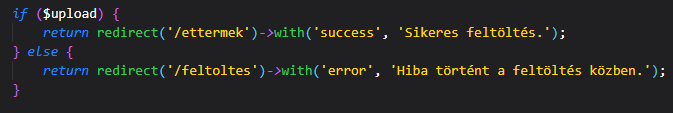
Ha a felhasználó minden kötelező adatot megadott, akkor utána a form értékeit a kontroller átkéri.



Miután változókba eltárolta az értékeket, a „*DB*” beépített osztállyal lefut egy feltöltés.

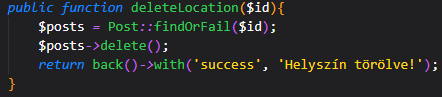


Ez a feltöltés egy változóba kerül eltárolásra, amit majd vizsgál, hogy sikeres volt e a feltöltés.

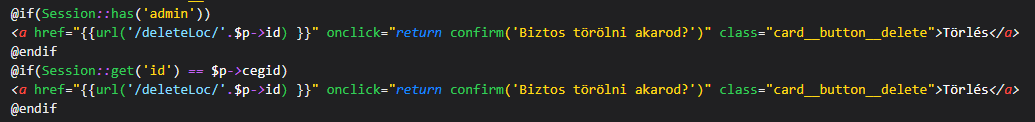


Ha sikeres a feltöltés, akkor az helyszínek oldalára fog visszairányítani, ha viszont nem sikerül a feltöltés, akkor a feltöltés oldalán marad, vagy laravel hibát dob.

## Helyszín törlése



A helyszín törlése ugyan úgy működik, mint a profil törlése, viszont azon a regisztrált cégek, amelyek feltöltötték a helyszínt, azok tudják a saját helyszíneiket törölni. Az admin mindet tudja törölni.



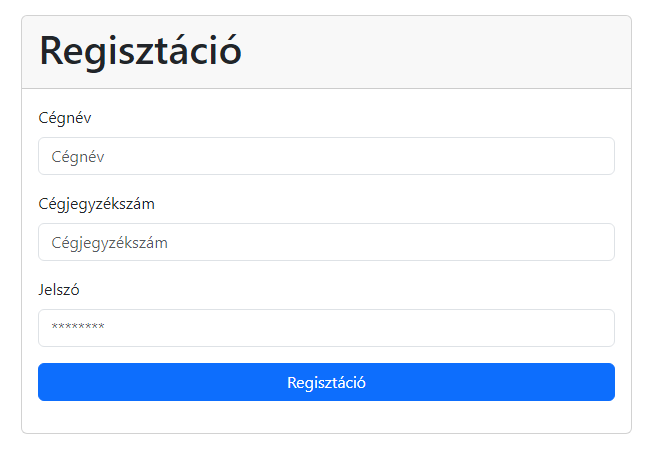
## Helyszín keresése



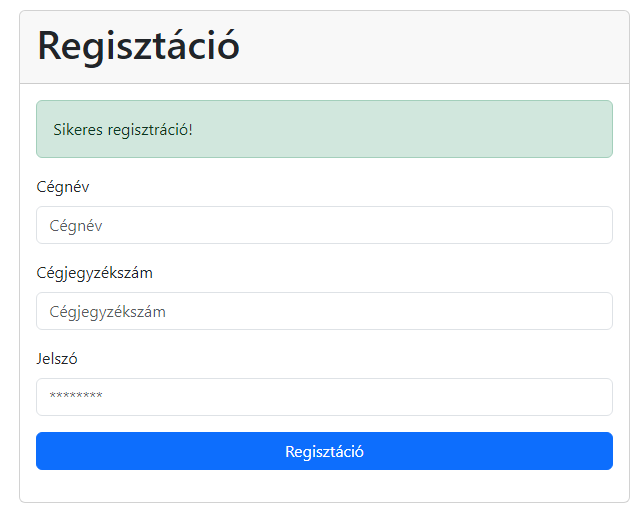
Helyszínt egy formon keresztül lehet keresni, majd annak az értékével futtat egy SQL lekérést. Ha nincs értéke a keresésnek, akkor az oldalt tölti be.

# Frontend

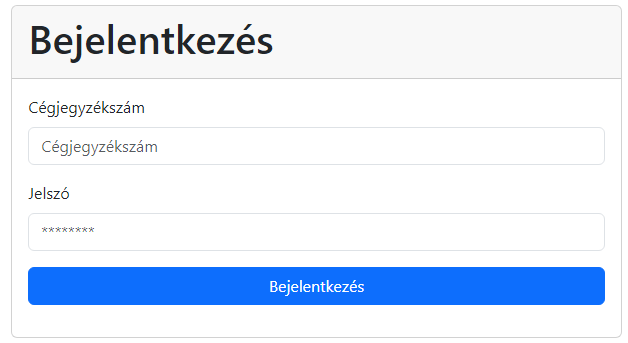
## Regisztráció nézet



A regisztrációs oldal egy egyszerű formból áll. Nevet, egy cégszámot és egy jelszót kér, majd ezeket meg az id -t, az időt amikor létre lett hozva és az időt amikor legutóbb módosítva lett hozzáadja az adatbázishoz, ha sikeres a regisztráció akkor egy zöld figyelmeztetés feljön a „cégnév” felett.



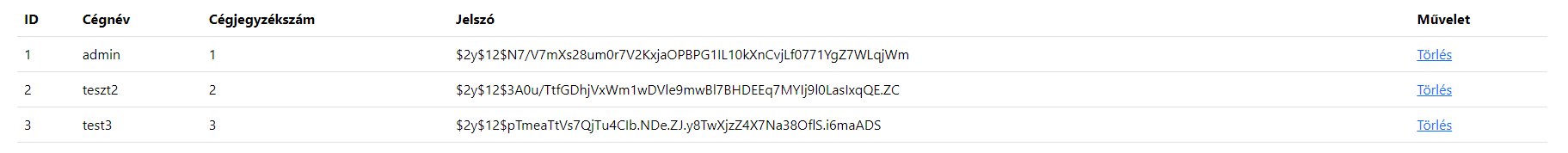
## Bejelentkezés nézet



A bejelentkezésnek is egyszerű kinézete van tesztelési indokokból. Itt ugyan úgy egy formot használ, majd ennek az adatait kapja meg a backend.

## Admin oldal

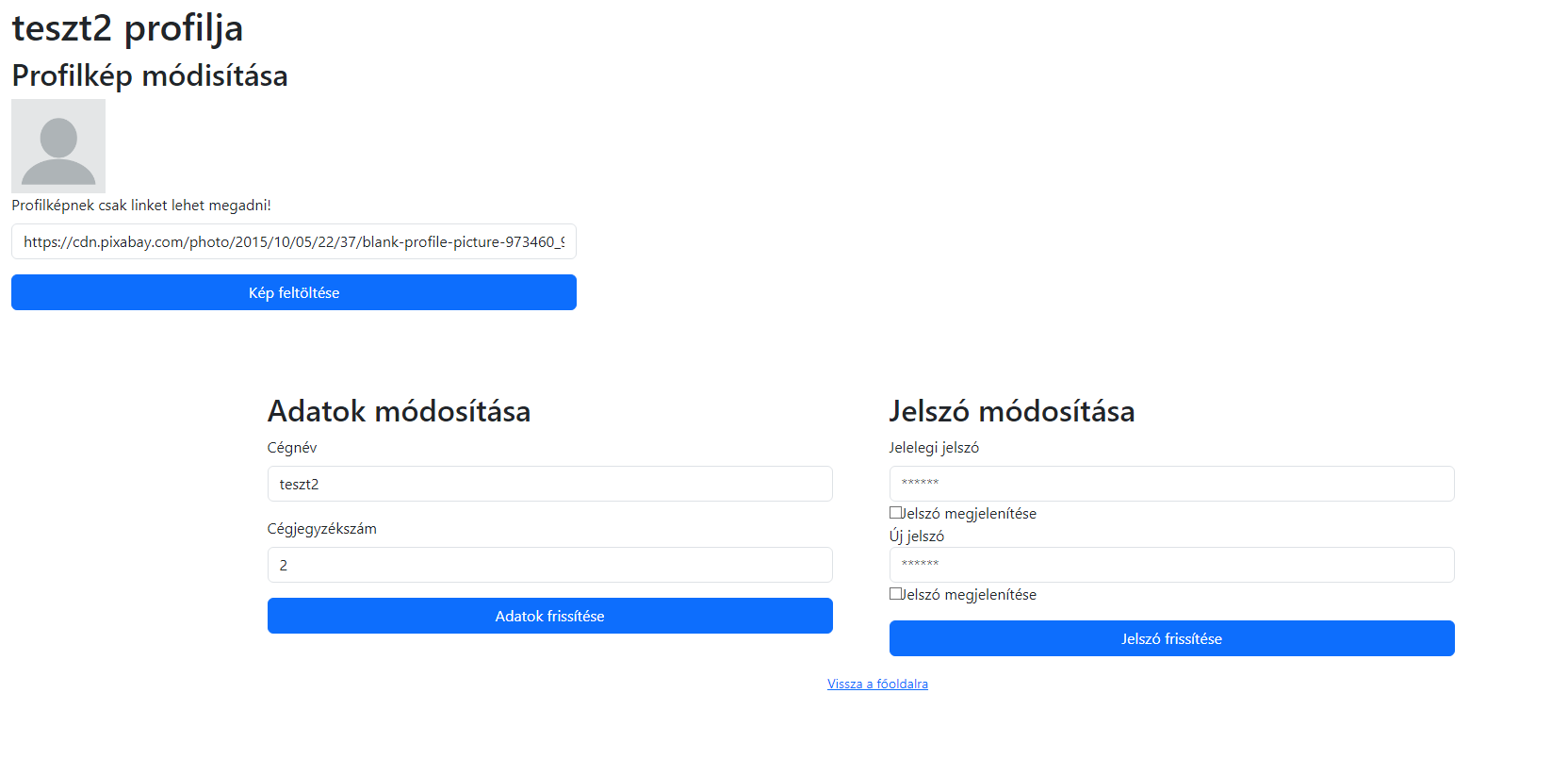
Egy működő admin oldal elengedhetetlen egy oldal számára, amin forgalom van, így itt is jelen van.



Az admin oldalon minden egyes regisztrált egyed megjelenik, itt lehet törölni is ezeket.

## Profil oldal

A profil oldal csak a cégeknek jelenik meg, ehhez nem tud az admin vagy látogató hozzáférni.



# Források

Korai register form logic: <https://www.youtube.com/watch?v=2MpZwFoBPjQ>

Korai php rest api: <https://www.youtube.com/watch?v=X51KOJKrofU>

Http válasz hiba kódok: [https://developer.mozilla.org/en-US/docs/Web/HTTP/Status#server\_error\_responses](https://developer.mozilla.org/en-US/docs/Web/HTTP/Status%23server_error_responses)

Laravel tutorial: <https://www.youtube.com/watch?v=ImtZ5yENzgE>

Adatok frissítése: több stackoverflow