

A tanár által létrehozott projekt klónoztuk ide: C:\xampp\htdocs\aa\laravel1\laravel\_test2

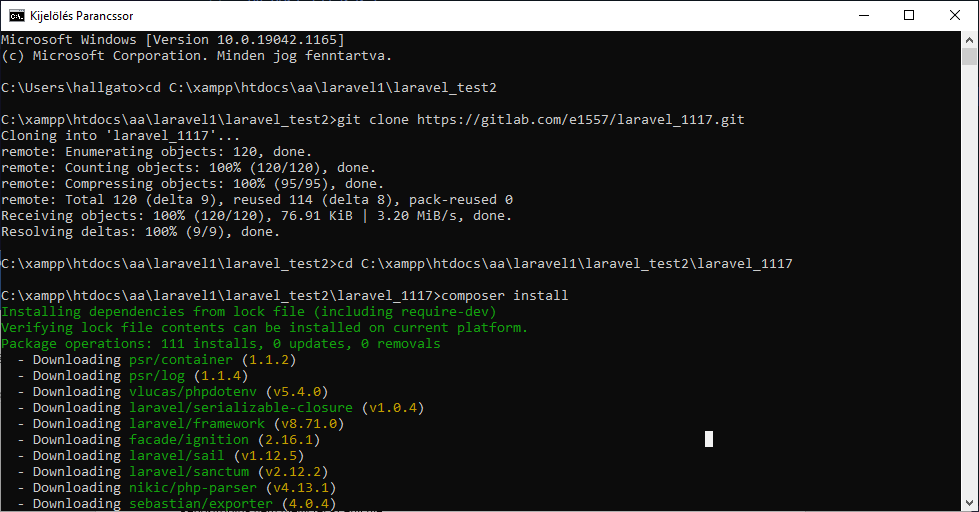
Repositoryba nem megy fel az env file!

Kulso fuggosegeket nem …

A leklónozott (cél) projekt folderére váltunk a cd paranccsal:

cd C:\xampp\htdocs\aa\laravel1\laravel\_test2\laravel1117

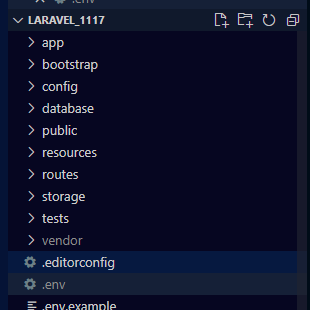
Kell egy *composer install* a célfolderből indítva 🡪 ekkor kerül sor az összes függőség telepítésére (composer.json –ban vannak a függőségek)



Egyedül a composer.json megy fel a repositoryba. A vendor mappa lesz az új.

env.example 🡪 a program milyen paramétereket vár

NAGYON FONTOS! Az .env.example file-ról CTRL+C és CTRL+V segítségével másolatot készítek, és átnevezem .env-re!



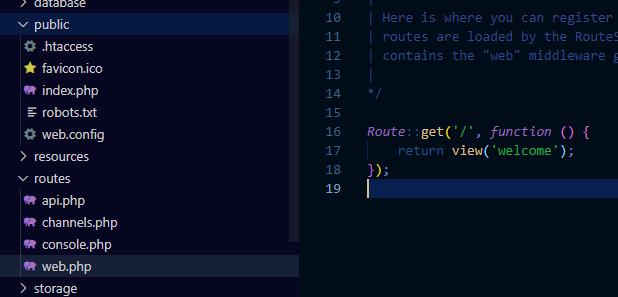
Új branchet hozok létre:



**Útvonal megadása**

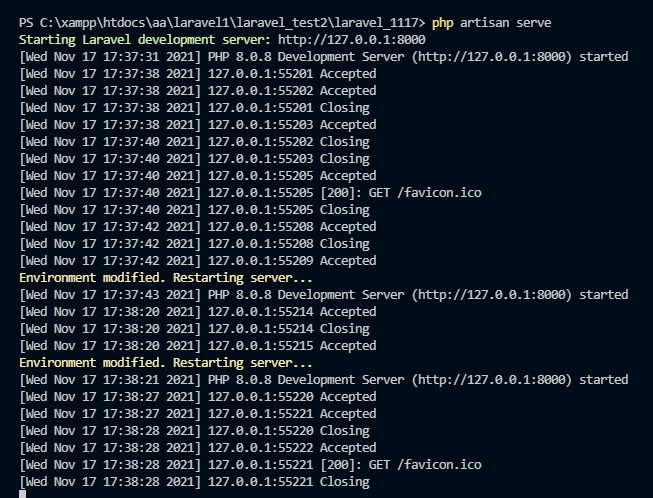
Csak a public mappa tartalmát hagyjuk elérni kívülről.

Web.php 🡪 Ez lesz az első belépési pont:

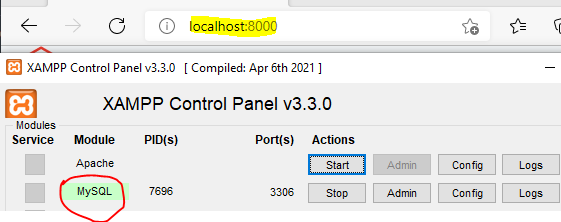


A *Route::get* utasítás vár egy útvonalat, és egy függvényt, amely visszatér valamivel.

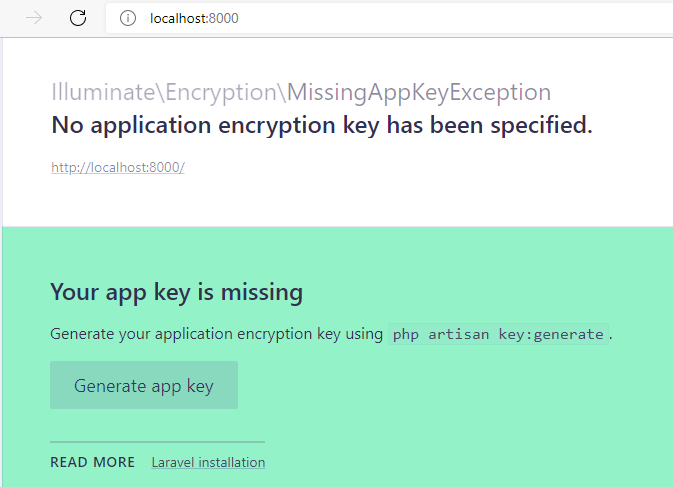
Powershell-re váltok: Terminal 🡪 New Terminal 🡪 Powershell

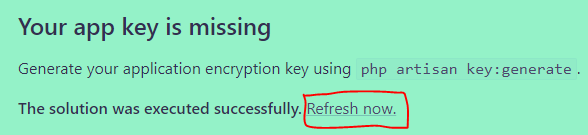


(A „Starting development server”-nél látjuk kiírva: <http://127.0.0.1:8000> , amely tul.képp ugyanaz, mint: <http://localhost:8000> . XAMPP-ban beindítom a MySQL-t, a böngészőbe pedig a linket: <http://localhost:8000>



A megljelenő oldalon: Generate App Key 🡪 Refresh Now:

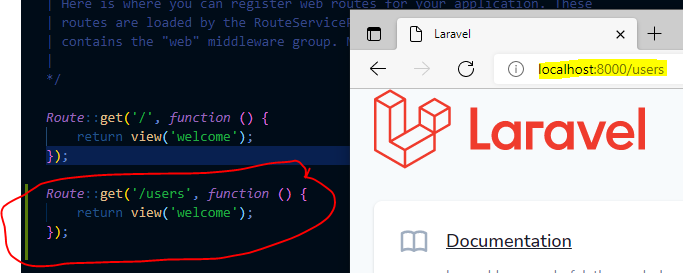




*Route*::get('/', *function* () {

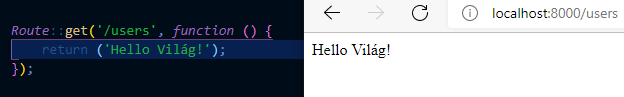
    return view('welcome');

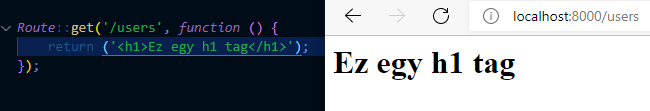
});



Tehát a ’/users’ rész kerül majd az URL végére, sikeresen beállítottuk az útvonalat.

Az osztály statikus függvényét hívjuk meg.

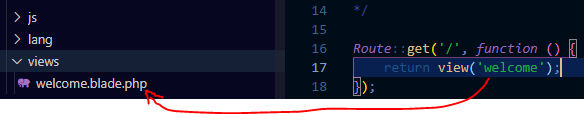




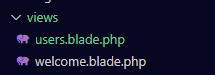
**View: a megjelenítés**

Ha a gyökér útvonalat adom meg, akkor a ()-be beírt tartalom jelenik meg.

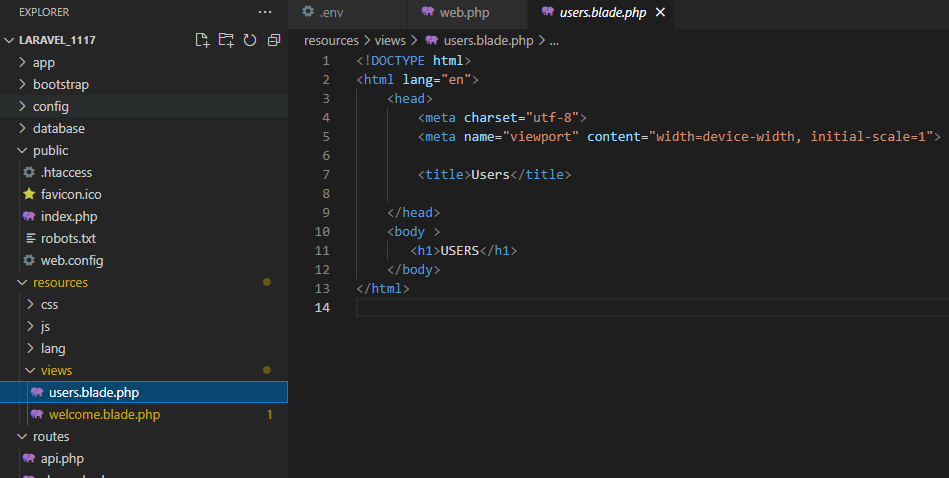
View-ek az erőforrás/resources alatt lesznek. A view-t senki nem fogja elérni kívülről, hanem áttranszformálódik a Public mappába, és a böngésző számára érthetővé válik.



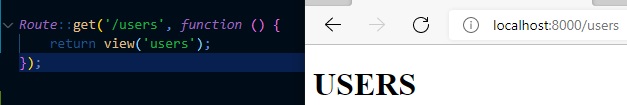
A Users view-ja:



Létrehozok egy users.blade.php-t, és beleírom a tartalmat, amelyet meg szeretnék jeleníteni a böngészőben:



a *view* függvény segítségével pedig megjelenítem a users.blade.php tartalmát:



„The Model is a PHP class that performs business logic and database manipulation like - retrieving, inserting, updating, and deleting data.”

**Esetünkben a modell: DB**

**Controller: összeköti a modellt és a view-kat.**

<https://laravel.com/docs/8.x/routing>

Nem view-t ad vissza, hanem a Controller megfelelő metódusát hívja meg. Előny: egy útvonal egy sor lesz:

use App\Http\Controllers\UserController;

Route::get('/user', [UserController::class, 'index']);

Útvonal: mit engedélyezünk, pl. mely útvonalakat…

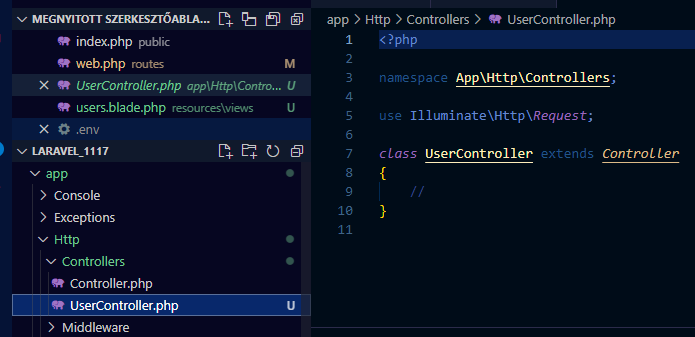
Controller: utasításokat ad, összefogja a nagyobb komponenseket

route::list 🡪 összes útvonal kilistázása

Controller létrehozása (példánkban a UserController nevet kapja):

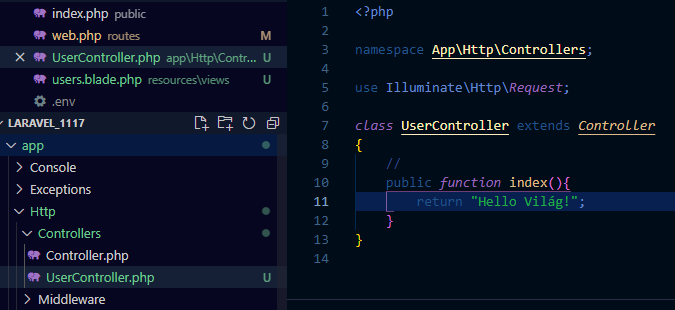
PS C:\xampp\htdocs\aa\laravel1\laravel\_test2\laravel\_1117> php artisan make:controller UserController

**Controller created successfully.**



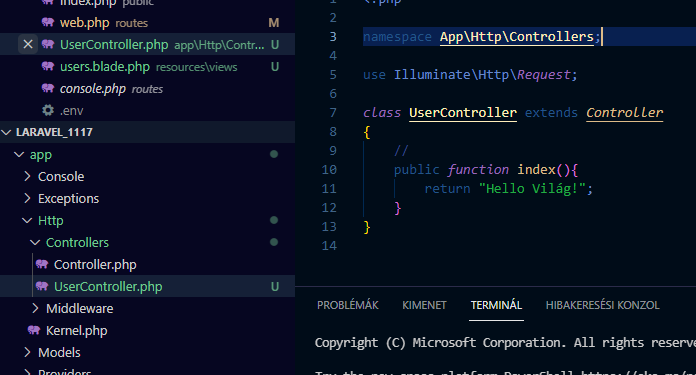
Controllerek: obj. osztályok, attribútumokkal és metódusokkal rend.

A Controller bizonyos függv. hívjuk meg



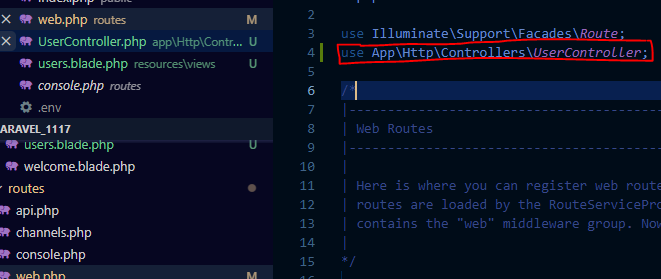
A *web.php* -ban definiáljunk egy útvonalat, amellyel elérjük az *UserConroller*-ben található *index* függvényt; ez a ’/user’ lesz:

*Route*::get('/user', [*UserController*::class, 'index']);

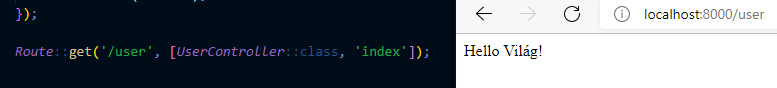


A *web.php* legfelső részén be kell importálni a *UserControllert* az alábbi formátumban:

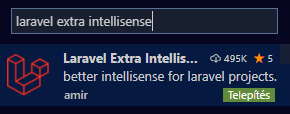
use App\Http\Controllers\UserController;

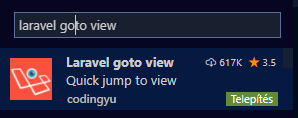


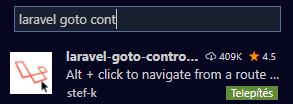
Az útvonal felhasználásával megjelenítjük a böngészőben a tartalmat. Láthatjuk, hogy az *index* függvény aktuális tartalma, a *„Helló Világ”* string szépen megjelent a böngészőben:



Az alábbi plug-inokat kell telepíteni:







**Útvonal létrehozásának lehetséges módjai**:

*Route*::get('/', *function* () {

     return view('welcome');

});

*Route*::get('/user1', *function* () {

    return 'Hello Vilag';

});

*Route*::get('/user2', *function* () {

    return view('users');

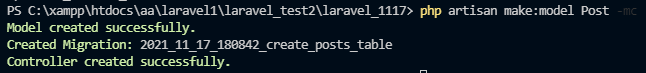
});

*Route*::get('/user3', [*UserController*::class, 'index']);

*Route*::get('/user4', [*UserController*::class, 'index2']);

Példa: Postokat szeretnénk létrehozni.

Migration, Controller, Model létrehozása egyben:



PS C:\xampp\htdocs\aa\laravel1\laravel\_test2\laravel\_1117> php artisan make:model *Post* -mc

Model created successfully.

Created Migration: 2021\_11\_17\_180842\_create\_posts\_table

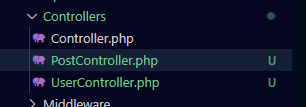
Controller created successfully.

A jobb panelből manuálisan kitöröljük a PostControllert, majd más módon hozzuk létre (az előző nem jó, előbb csak a controllert kell létrehozni):

PS C:\xampp\htdocs\aa\laravel1\laravel\_test2\laravel\_1117> php artisan make:controller PostController -m Post

Controller created successfully.

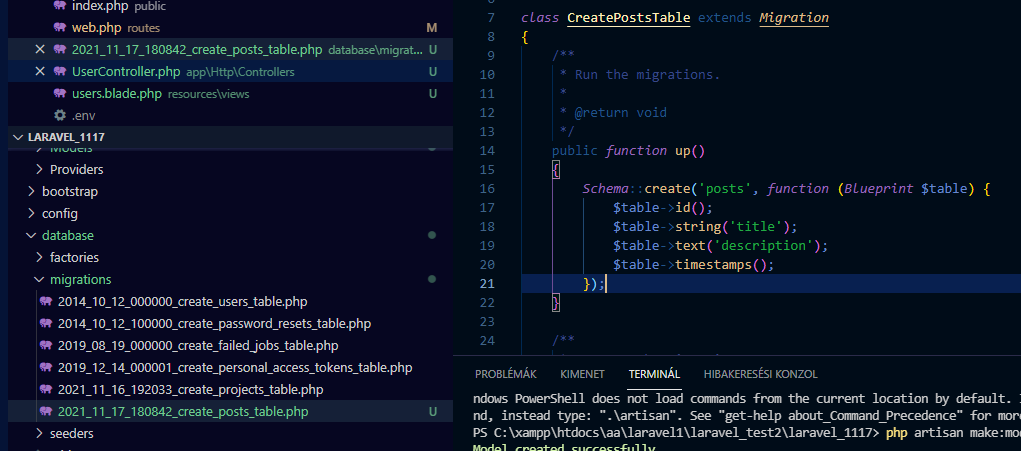




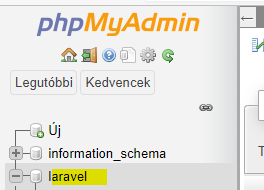
Ha a modellt hozzuk létre, akkor ő létrehozza a migrationt a konvenciónak megfelelően

php artisan make:model Post -m

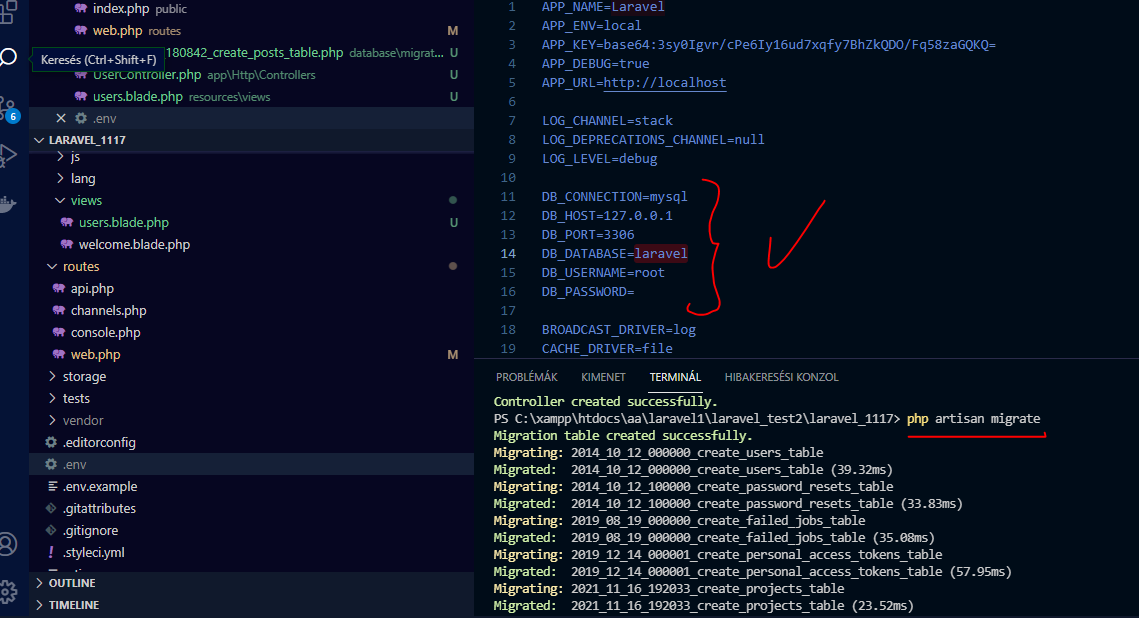
database 🡪 migrations 🡪 2021\_11\_17\_180842\_create\_posts\_table.php 🡪 a public function up() alatt létrehozzuk a táblaattribútumokat:



Kreálunk egy *laravel* nevű adatbázist phpMyAdmin-ban:



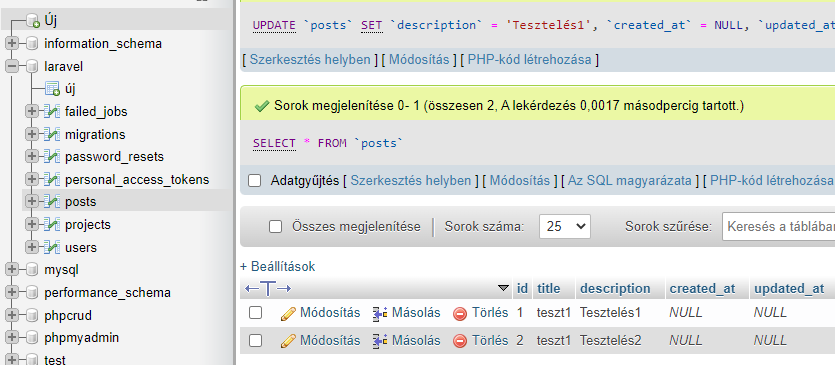
Ellenőrizzük, ha kell, korrigáljuk a kapcsolati adatokat az .env fájlban:



Futtatjuk a migrációt:

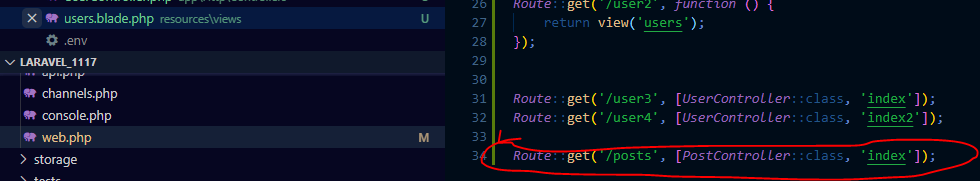
php artisan migrate

A migrációs tábla létrejött, beleértve a *posts* táblát is a létrehozott attribútumokkal, melyet feltöltünk néhány tesztadattal, melyek a postokat fogják szimbolizálni:

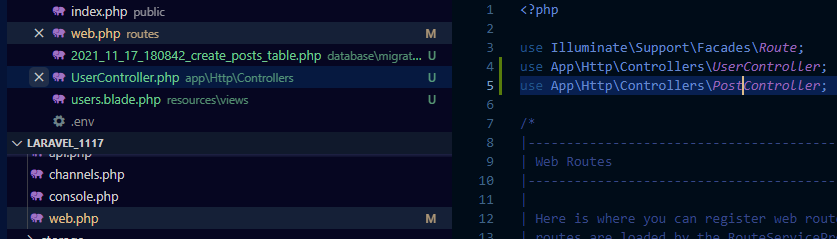


**Adatok lekérdezése**

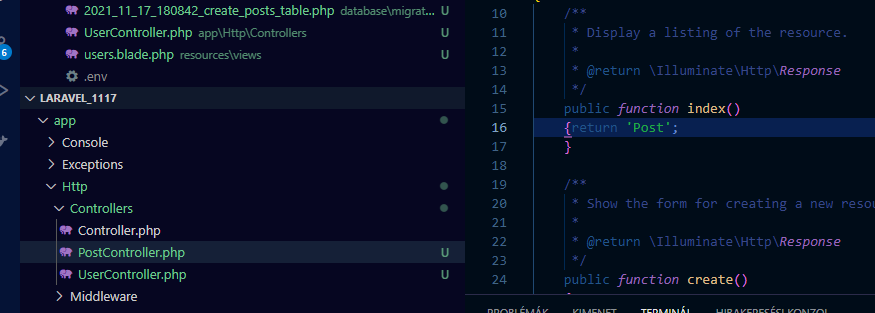
Először is kell egy útvonal.

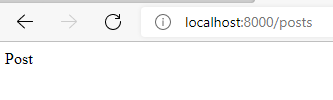


A *web.php*-ba beimportáljuk a PostControllert:



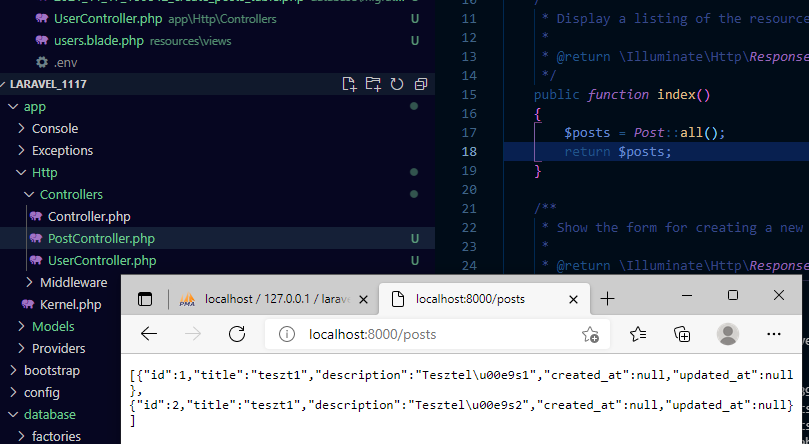
Összekötjük, majd teszteljük, hogy működik-e a kapcsolat:





Bővítjük az index függvényt. Bevezetünk egy *$posts* változót a lekérdezés eredményének tárolásához.

*all()* függvény: az adatbázisban található postok közül kiszedi az összeset



Ha konkrétan JSON formátumban szeretnénk visszakapni az eredményt:

 public *function* index()

    {

        $posts = *Post*::all();

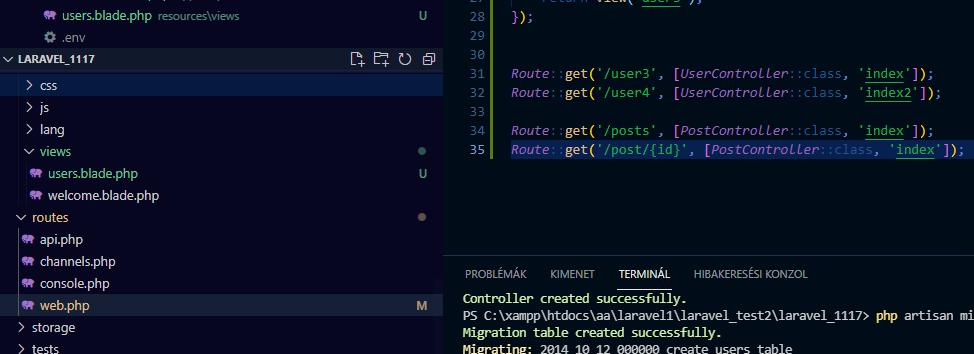
        return response()->json($posts);

        return $posts;

    }

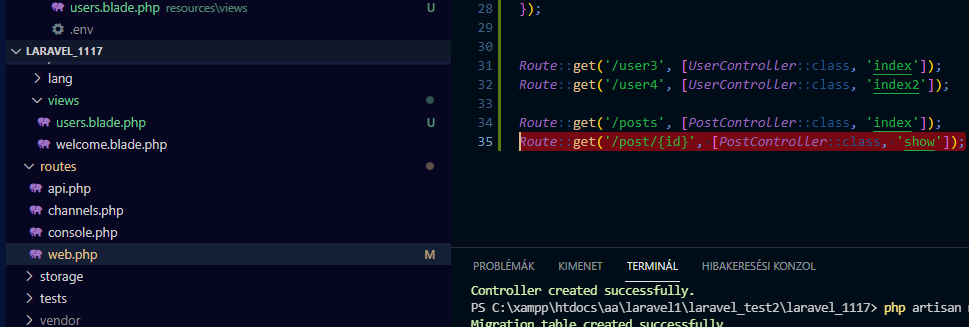
A *web.php* alatt definiálunk egy új útvonalat (’/post/{id}’), amely az URL végére beírt indexű postot fogja visszaadni:

*Route*::get('/post/{id}', [*PostController*::class, 'index']);

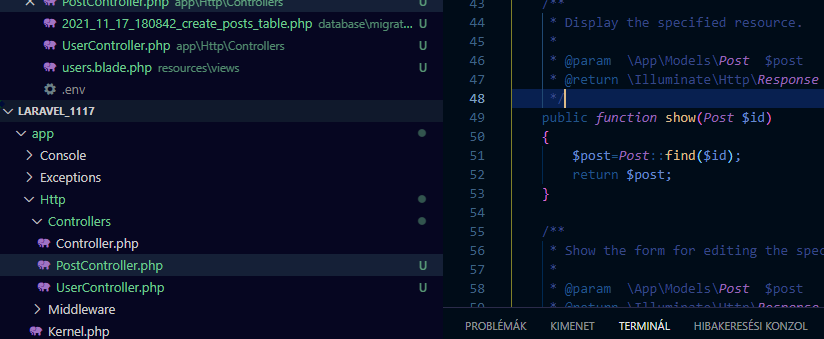


Ezzel az útvonallal meghívjuk a PostController *show* függvényét:

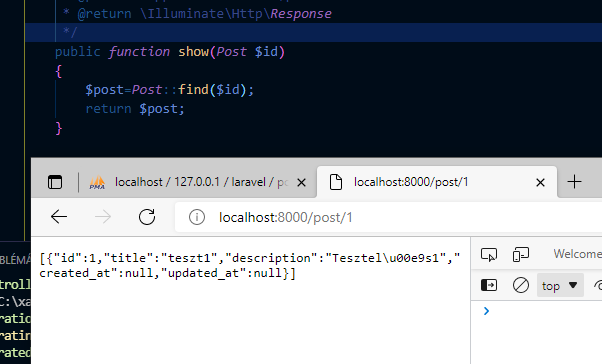
*Route*::get('/post/{id}', [*PostController*::class, 'show']);



Bővítjük a show() függvény tartalmát. Lesz egy (Post típusú) bemenő $id paraméterünk, amely a keresett post indexét szimbolizálja. A find() utasítás pedig megkeresi az adott indexű postot.



Sikerült tehát az 1-es indexű post adatainak kiiratása:



{

        $post=*Post*::findOrFail($id);

        return $post;

    }

public *function* show( $id)

    {

        //$post=Post::findOrFail($id);

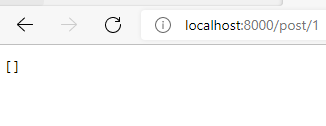
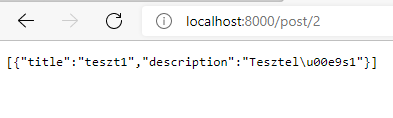
        $posts=*Post*::where('id','<',$id)->

        select('title','description')->

        get();

        return $posts;

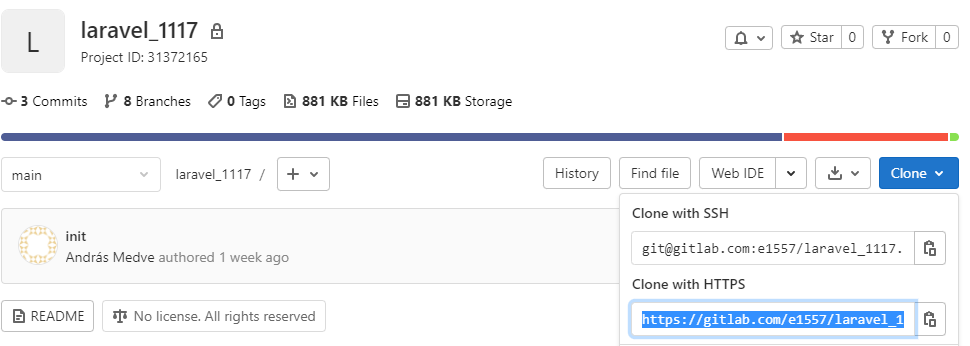
    }

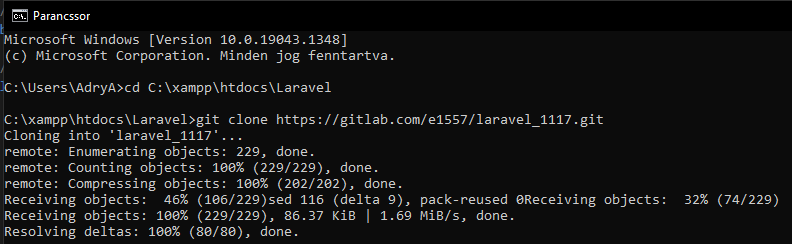
 

1. **find($id)** takes an id and returns a single model. If no matching model exist, it returns *null*.
2. **findOrFail($id)** takes an id and returns a single model. If no matching model exist, it throws an error1.
3. **first()** returns the first record found in the database. If no matching model exist, it returns *null*.
4. **firstOrFail()** returns the first record found in the database. If no matching model exist, it throws an error1.
5. **get()** returns a collection of models matching the query.
6. **pluck($column)** returns a collection of just the values in the given column. In previous versions of Laravel this method was called lists.
7. **toArray()** converts the model/collection into a simple PHP array.

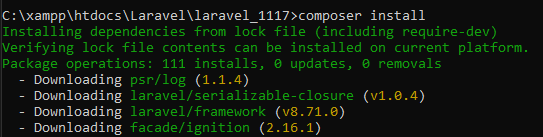
Megjegyzés:

ha otthon le akarom klónozni a saját branchemet:

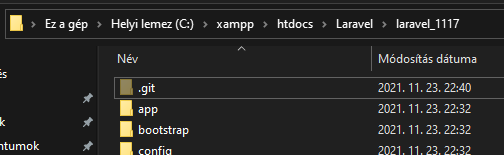




Nagyon fontos lefuttatni *a composer install*-t!



Sikerült leklónozni a main branch-et:



Behívom a projectet Visual Studio Code-ba, és legalul a main-re kattintok, majd a legördülő listából kiválasztom a saját branchem.

