Backend építése Laravelben

## Összes eddigi tananyag felfűzve [itt](https://docs.google.com/document/d/1ybnWk0tEwyhRDPbPCpyBNf4YEybYqK1jbMC2fahuEJ4/edit?usp=sharing) található (folyamatosan bővül).

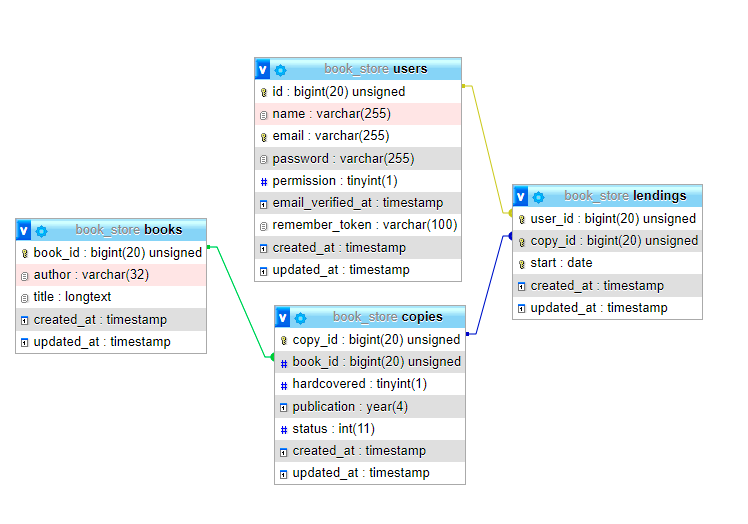
## Könyvtár adatbázis bővítése

Kiindulási fájl lehet az, amit a dolgozat előtt elkészítettél (bejelentkezéssel és regisztrációval), de ha nincs meg, [innen](https://github.com/Agnes-milia/book_with_breeze.git) is letöltheted a kiinduló Laravel projektet.

## Kölcsönzés (Lending) - lendings tábla beépítése

Az alap elgondolás az, hogy a kölcsönzés szétválasztható a példánytól, így a copies táblában más hasznos tulajdonságokat is tárolhatunk (kemény kötésű-e, kiadás éve), ami felveti az [oszlop típusainak újabb használatait](https://laravel.com/docs/9.x/migrations#available-column-types). Így létrehozunk egy lendings nevű táblát (Model, majd Controller létrehozása Lending, illetve LendingController névvel), amiben a kölcsönzés tulajdonságait tárolhatjuk (példány, felhasználó, kölcsönzés kezdete…).

*Találós kérdés:* az eddigi státus oszlopot hova tennéd?



*Emlékeztető (ismétlés)*:

### Azonosítók beállítása[[1]](#footnote-0)

Ha egy táblában nem az id az alapértelmezett azonosító (más nevet szeretnénk), akkor minden Model classben át kell írni a primary key-t így:

protected $primaryKey = 'task\_id';

és akkor működik az sql parancs...

Ha nem tesszük, akkor a find és egyéb fg-ek az id nevű primary keyt keresik.

### Időbélyeget nem kérek…

Amennyiben nem szeretnéd használni a timestamp() függvényt, ami rögzíti, hogy mikor lett létrehozva, módosítva egy rekord, akkor a Modelben írd az alábbiakat:

public $timestamps = false;

### Többszörös elsődleges kulcs létrehozása egy táblában (migrációs fájlban):

$table->primary(['user\_id', 'avatar\_id', 'task\_range\_id']);

## **Admin, illetve egyéb más autentikációs réteg létrehozása**

1. Réteg (middleware) létrehozása: php artisan make:middleware IsAdmin

2. majd belépünk a most létrehozott fájlba (app\Http\Middleware\IsAdmin.php), és átírjuk a handle függvényt pl. így:

public function handle($request, Closure $next) { if (Auth::user() && Auth::user()->permission == 0) { return $next($request); } return redirect('home')->with('error','You have not admin access'); }

3. ugyanitt fent írjuk be ezt is: use Illuminate\Support\Facades\Auth;

4. majd belépünk a már létező app/Http/Kernel.php fájlba, és kiegészítjük a $routeMiddleware lista elemét az alábbi sorral (pl. az auth után):

'admin' => \App\Http\Middleware\IsAdmin::class,

5. útvonalak beállítása a jogkörök mentén:

Route::middleware( ['admin'])->group(function () {

Route::apiResource('/users', UserController::class);

});

6. blade fájlok (opcionális) és

7. controllerek (opcionális) aktualizálása

**Unit tesztek** mellékelve az etananyagon (importálható a Thunderben)…

## Validálás[[2]](#footnote-1)

Próbáld ki és teszteld az alábbikódrészletet a UserControllerben (Thunderben tesztelheted):

public function updatePassword(Request $request, $id)

{

$validator = Validator::make($request->all(), [

"password" => 'string|min:3|max:50',

]);

if ($validator->fails()) {

return response()->json(["message" => $validator->errors()->all()], 400);

}

$user = User::where("id", $id)->update([

"password" => Hash::make($request->password),

]);

return response()->json(["user" => $user]);

}

1. hasonló szavak kijelölése: Ctrl+Shift+L [↑](#footnote-ref-0)
2. <https://laravel.com/docs/9.x/validation#quick-writing-the-validation-logic> [↑](#footnote-ref-1)