**Algemeen:**

Migratie = met een migratie kan je een database table aan laten maken.

Seeder = met een seeder kan je een database table invullen / laten invullen met gegevens.

**Terminal comments:**

cd .. = een keer terug in de file structure bijvoorbeeld: van /C/Users naar /C .

cd $name = voeg dit toe aan de file structure bijvoorbeeld: van /C naar /C/$name.

composer create-project laravel/laravel $nameProject = Maakt een Laravel project.

php artisan make:controller $controllerName = maakt een controller.

php artisan make:controller $controllerName -– resource = maakt een controller aan met de algemenen CRUD functies er al in.

Php artisan make:model modelName(enkelvoud) -m = maakt een model met de modelName en maakt er meteen een migration bij.

**CSS**

npm install = als je dit installeer kan je je css opnieuw compilen.

npm run dev = zorgt ervoor dat je de compiled css file opnieuw compiled.

npm run watch = de compile function blijft doorgaan en als hij een aanpassing ziet voor hij deze meteen door.

**Migration**

php artisan migrate = voert alle migrations uit.

php artisan migrate:refresh = Verwijdeje al je migrations en maakt hem weer zoals hij toen was. Verwijderd niet alle tables.

php artisan migrate:refresh –seed = Het zelfde als een normale refresh maar deze voert ook meteen alle seeds uit.

Php artisan migrate:fresh = Verwijdeje al je migrations en maakt hem weer zoals hij toen was. Verwijderd wel alle tables.

php artisan route:list = laat alle routes zien die er op dit moment zijn.

**Seeders**

php artisan make:seeder $NameTableSeeder = maakt een seeder.

php artisan db:seed = voert de seed uit.

php artisan db:seed --class=UserSeeder = Het zelfde als db:seed alleen kies je hier een specifieke class.

composer dump-autoload = Dit is nodig om de seeders te laten werken. Dit moet elke keer gebeuren als je iets aanpast, meerdere maakt of als je er één verwijderd.

Php artisan DB::rollBack = maakt de migrations en seeds opnieuw.

**Database:**

Inlog gegevens [

gebruikersNaam = root,

wachtwoord = mysql

];

Database tables zijn altijd meervoud

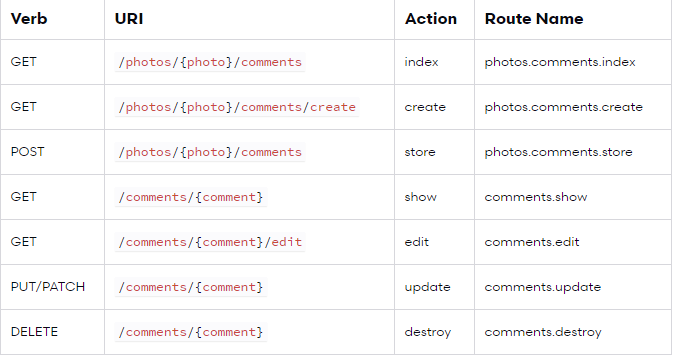
Modus zijn altijd enkelvoud

Controllers zijn altijd / bijna altijd meervoud

Controller namen altijd HoofdletterPerWoord

**Routing**

Route::resource('photos.comments', 'CommentsController'); = je moet eerst in je url photos hebben staan die naar 1 hebben laten linken en dan kan je comments gebruiken om bij de comments controller te komen.



**.env**

{{config(“app.name”)}} = APP\_NAME (Zo kan je gegevens van de .env file halen)

**Shopping cart workshop**

Zorg dat je alle shopping cart functions in één class hebt, zodat een front-end dev hier niks mee hoeft te doen.

Zorgt dat je eerst weet hoe je iets gaat maken en om het makkelijker te maken maak alvast functions met names en vars om het logischer te maken voor jezelf of andere hoe het te werk gaat.