Laravel 9 – Filament – Memory Sheet

Hello!

My name is **Martin**. I am a PHP developer from Germany with 20+ years of E-Commerce / B2B experience.

Since I started to use the framework **Laravel** for PHP development, my programming experience and the results have **improved dramatically**.

I want to share this experience with you because I think that our world is in need for high-quality and user-friendly websites.

But first, let's take care of business! :-)

In this course, we start to create a real estate investment website. In this very profitable business area, I am thinking about a B2B platform with some social elements where global investors can find, reserve and acquire promising real estate assets.

During this course part, we will create a **backend** with Laravel **Filament** where the admins of the company can manage the real estate assets that are displayed in the frontend.
For the management of the images, we will use a famous plugin that is called “Media Library”. This is something you have to know for successful Laravel development.

I am aware that many of you guys have just finished university or other schools and would like to get an interesting and well-paid job in the programming business.
And you will!
PHP with Laravel is a top-notch technology that will give you a great start into website programming.

Enough said. Let's start and have some fun!

Martin

Project Setup

Laravel Installation

composer create-project laravel/laravel:VERSION PROJECT-NAME

https://laravel.com/docs/9.x/installation#your-first-laravel-project

---

Jetstream Installation

composer require laravel/jetstream

php artisan jetstream:install livewire

npm install

npm run build

php artisan migrate

php artisan vendor:publish --tag=jetstream-views

Docs: https://jetstream.laravel.com/2.x/installation.html

“Migrate-Fresh resistant” admin user

/database/seeders/DatabaseSeeder.php

```
\App\Models\User::factory()->create([
    'name' => 'Martin',
    'email' => 'martin@laravel-php.com',
    'password' => bcrypt( value: 'bla123')
]);
\App\Models\User::factory( count: 10)->create();
```

Continue next page

Filament Installation

Installation: `composer require filament/filament:"^2.0"`
Configuration: `php artisan vendor:publish --tag=filament-config`
Translations: `php artisan vendor:publish --tag=filament-translations`

For **Upgrades** in **composer.json**:

```
"post-update-cmd": [
  "@php artisan filament:upgrade"
],
```

Docs: https://filamentphp.com/docs/2.x/admin/installation#installation

---

TailwindCss color palette

https://tailwindcss.com/docs/customizing-colors#default-color-palette

---

Continue next page

Constants in Laravel

Constants you can define in the ***/config*** folder in files, e.g., ***/config/company.php***

In this **config files**, you just return an **array** that also can contain **nested arrays** within them.

/config/company.php

```php
return [
    'name' => 'Real Invest',

    'address' => [
        'city' => 'New York',
        'street' => '111 Main St.'
    ]
];
```

Nested array

Accessing the constants with the config function:

/resources/views/startpage.php (later moved to */resources/views/pages/startpage.php*)

```
a">Street .</span> {{ config('company.address.street') }}
">
```

File name in config folder

Array Key

Array key in nested Array

Continue next page

Vendor:publish for HTML Editing of Filament

`php artisan vendor:publish`

Choose the correct number on the right and hit ENTER…

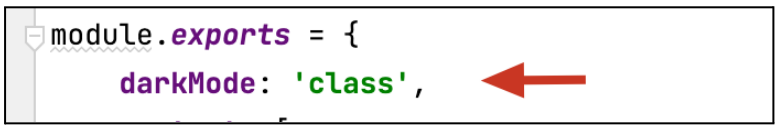
To insert the logo, we edited:
*/resources/views**/vendor/filament/components/header/index.blade.php***

Create a filament theme

Follow instructions here:
https://filamentphp.com/docs/2.x/admin/appearance#building-themes

Tips:

- **Leave out** everything that has to do with "**Mix**" (old bundler)
- Remember to create "**filament.css**"
- To "re-enable" **dark mode**, put this in the ***tailwind.config.js***:



About dark mode, read here: https://tailwindcss.com/docs/dark-mode#basic-usage

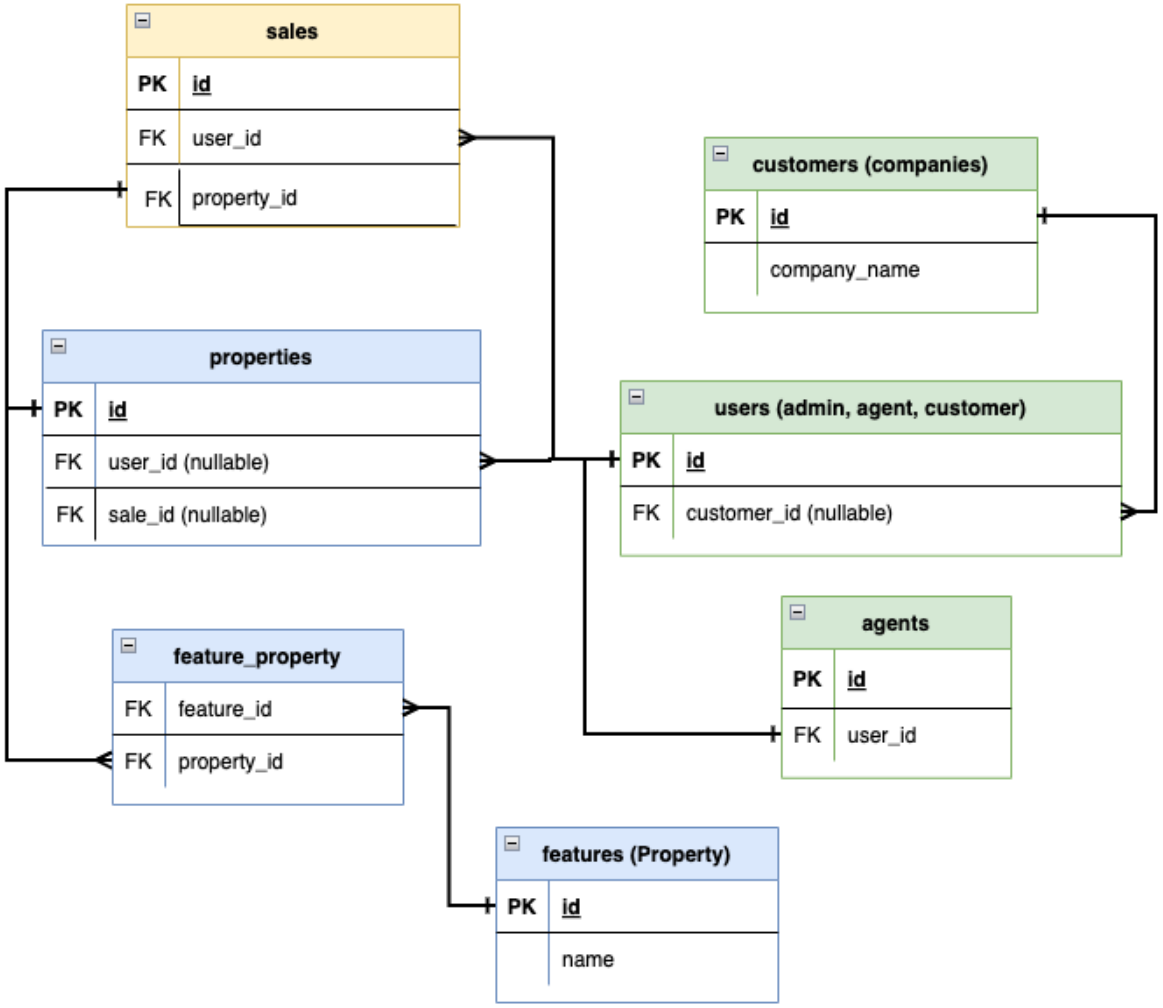
---

Continue next page

Building Backend App with Laravel Filament

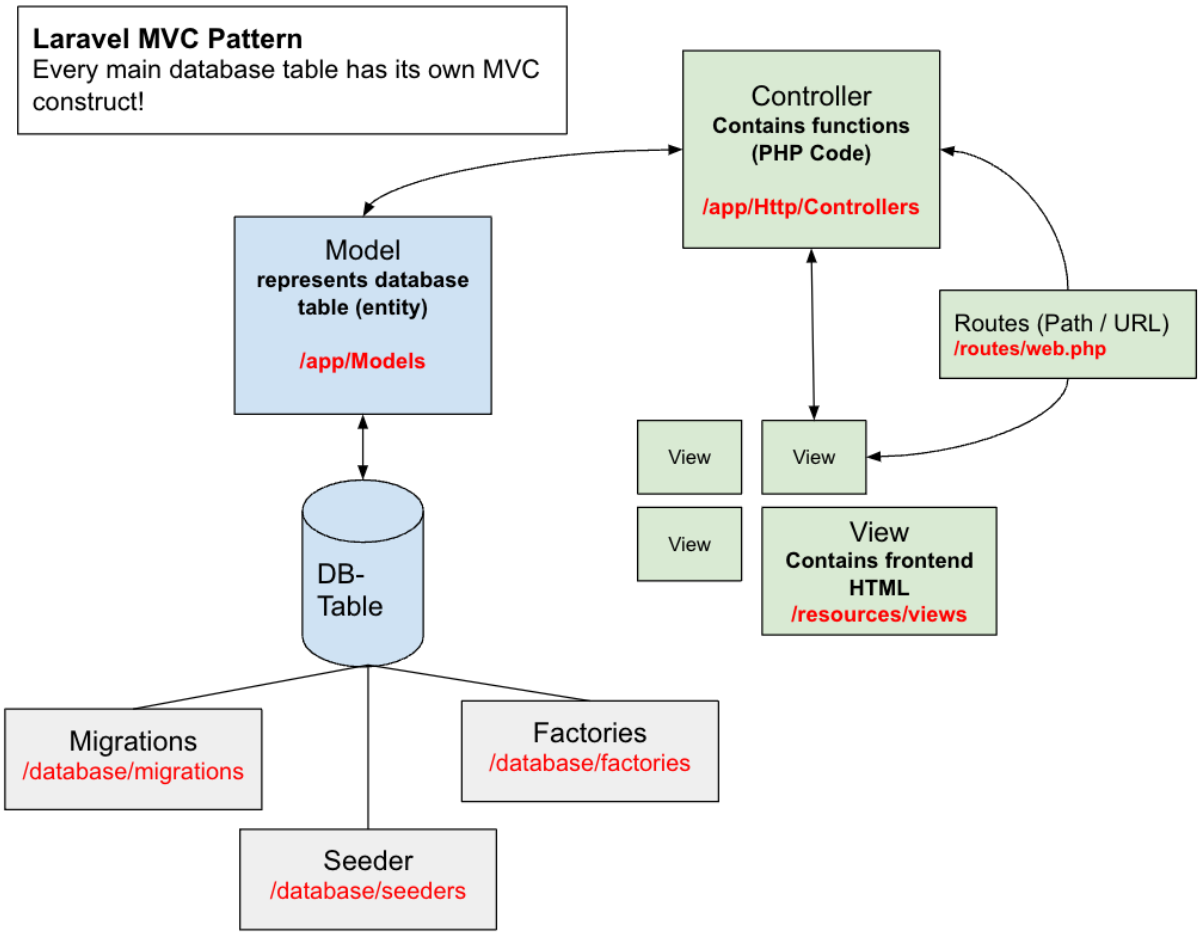
Database Diagram

Database draft, done with ***drawio* plugin** of Google Docs



Continue next page

MVC Pattern in Laravel



Create Model with **m**igration (for DB), **f**actory, and **c**ontroller

`php artisan make:model Property -mfc`

For **model name**, always use **English, Singular, Capital letter** at the beginning.

Fill in fields. Use **users migration** as template, or look for field types in docs:
https://laravel.com/docs/9.x/migrations#available-column-types

---

Continue next page

Soft-Deletes (Deletions in a way that you can restore later)

In Model */app/Models/Propery.php*

```php
use Illuminate\Database\Eloquent\SoftDeletes;

class Property extends Model
{
    use HasFactory, SoftDeletes;
```

In Migration File:
$table->softDeletes();

Migrate into database: php artisan make:migrate

Factory, Seeding, and Filament backend Resource

Use a Factory with Faker

/database/factories/ProperyFactory.php

```php
public function definition()
{
    return [
        'title' => $this->faker->realTextBetween( minNbChars: 25, maxNbChars: 45),
        'description' => $this->faker->realTextBetween( minNbChars: 100, maxNbChars: 150),
        'country' => $this->faker->country(),
        'city' => $this->faker->city(),
        'address' => $this->faker->address(),
        'price' => rand(500, 5000)*1000,
```

As you can see, the **Faker Library** can create all kinds of fake data.

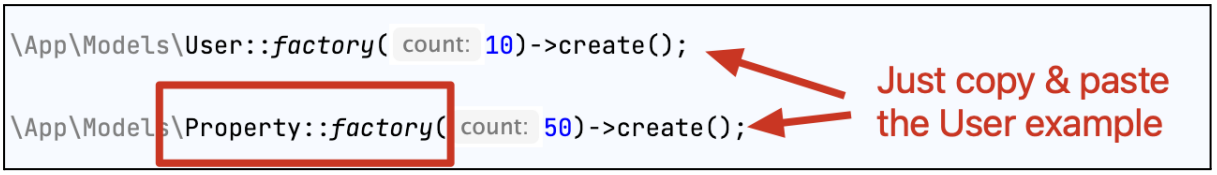
If you want to see **everything** that faker offers, just look here:
https://github.com/fzaninotto/Faker

The **rand()** function of PHP you can use for **random integer** numbers.

Important Tip: If you do not know where to start,
just use ***/database/factories/UserFactory.php*** as a template.
This file is part of **every** Laravel **installation**.

Continue next page

Call Factory from */database/DatabaseSeeder.php*



Creating your first Filament resource

Always do this first: **composer require doctrine/dbal**

Docs:
https://filamentphp.com/docs/2.x/admin/resources/getting-started#automatically-generating-forms-and-tables

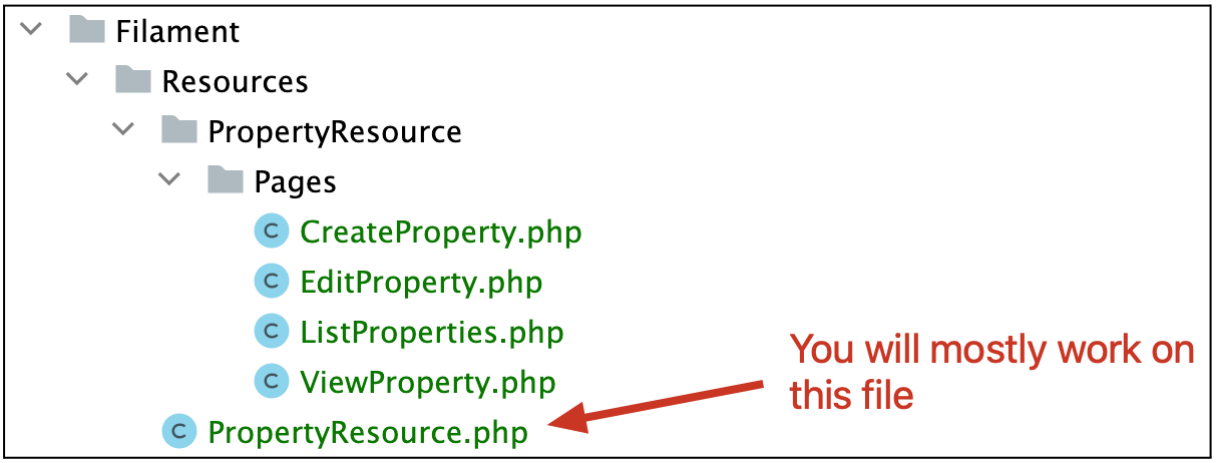
Now, you can **Auto-Generate** the resource with **complete table and fields** like this:

`php artisan make:filament-resource MODEL NAME --generate`

You can also use the `--soft-deletes` and the `--view` options, so a complete example would look like this:

`php artisan make:filament-resource Property --generate --soft-deletes --view`

Now you can **find** the **new resource** here:

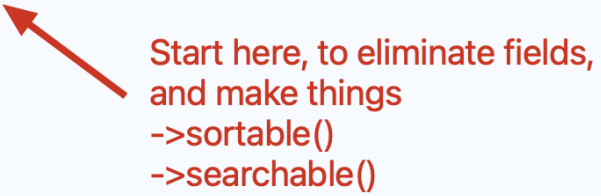


Continue next page

/app/Filament/Resources/ YOUR-RESOURCE.php

```php
public static function form(Form $form): Form
{
    return $form
        ->schema([...]);
}

public static function table(Table $table): Table
{
    return $table
        ->columns([...])
        ->filters([...])
        ->actions([...])
        ->bulkActions([...]);
}
```



Eliminate all columns that are not so relevant in the list view of the resource, do some first **column adaptations**, like:

→sortable()	→searchable()	→limit()
→label()	→money()	→default()
→alignLeft() / Right / Center		

This document will be continued with the rest of the course finished….