nstallation

Requirements

ament requires the following to run:

- PHP 8.1+
- Laravel v10.0+
- Livewire v3.0+

Installation

If you are upgrading from Filament v2, please review the upgrade guide.

stall the Filament Panel Builder by running the following commands in your Laravel project director

```
composer require filament/filament:"^3.2" -W

php artisan filament:install --panels
```

is will create and register a new Laravel service provider called `p/Providers/Filament/AdminPanelProvider.php`.

If you get an error when accessing your panel, check that the service provider was registered in `bootstrap/providers.php` (Laravel 11 and above) or `config/app.php` (Laravel 10 and below). If not, you should manually add it.

Create a user

u can create a new user account with the following command:

```
php artisan make:filament-user
```

ɔen `/admin` in your web browser, sign in, and start building your app!

ɔt sure where to start? Review the Getting Started guide to learn how to build a complete Filament min panel.

Using other Filament packages

e Filament Panel Builder pre-installs the Form Builder, Table Builder, Notifications, Actions, olists, and Widgets packages. No other installation steps are required to use these packages with ɔanel.

Improving Filament panel performance

Optimizing Filament for production

optimize Filament for production, you should run the following command in your deployment scrip

```
ɔhp artisan filament:optimize
```

is command will cache the Filament components and additionally the Blade icons, which can ɟnificantly improve the performance of your Filament panels. This command is a shorthand for the mmands `php artisan filament:cache-components` and `php artisan icons:cache`.

clear the caches at once, you can run:

```
ɔhp artisan filament:optimize-clear
```

Caching Filament components

ʏou're not using the `filament:optimize` command, you may wish to consider running `php artisan lament:cache-components` in your deployment script, especially if you have large numbers of mponents (resources, pages, widgets, relation managers, custom Livewire components, etc.). Thi ll create cache files in the `bootstrap/cache/filament` directory of your application, which contain dexes for each type of component. This can significantly improve the performance of Filament in me apps, as it reduces the number of files that need to be scanned and auto-discovered for mponents.

›wever, if you are actively developing your app locally, you should avoid using this command, as it ll prevent any new components from being discovered until the cache is cleared or rebuilt.

u can clear the cache at any time without rebuilding it by running `php artisan filament:clear-` `:hed-components`.

Caching Blade Icons

ʻou're not using the `filament:optimize` command, you may wish to consider running `php artisan` `ons:cache` locally, and also in your deployment script. This is because Filament uses the Blade Icoı ckage, which can be much more performant when cached.

Enabling OPcache on your server

›m the Laravel Forge documentation:

> **Optimizing the PHP OPcache for production will configure OPcache to store your compiled PHP code in memory to greatly improve performance.**

ease use a search engine to find the relevant OPcache setup instructions for your environment.

Optimizing your Laravel app

u should also consider optimizing your Laravel app for production by running `php artisan optimiz` your deployment script. This will cache the configuration files and routes.

Deploying to production

Allowing users to access a panel

default, all `User` models can access Filament locally. However, when deploying to production or ıning unit tests, you must update your `App\Models\User.php` to implement the `FilamentUser` ntract — ensuring that only the correct users can access your panel:

```
:?php

ıamespace App\Models;

ıse Filament\Models\Contracts\FilamentUser;
ıse Filament\Panel;
```

```
use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable implements FilamentUser
{
    // ...

    public function canAccessPanel(Panel $panel): bool
    {
        return str_ends_with($this->email, '@yourdomain.com') && $this->hasVerifiedEmail();
    }
}
```

If you don't complete these steps, a 403 Forbidden error will be returned when accessing the app in production.

arn more about users.

Using a production-ready storage disk

ament has a storage disk defined in the configuration, which by default is set to `public`. You can t the `FILAMENT_FILESYSTEM_DISK` environment variable to change this.

e `public` disk, while great for easy local development, is not suitable for production. It does not pport file visibility, so features of Filament such as file uploads will create public files. In productio u need to use a production-ready disk such as `s3` with a private access policy, to prevent authorized access to the uploaded files.

Publishing configuration

u can publish the Filament package configuration (if needed) using the following command:

```
php artisan vendor:publish --tag=filament-config
```

Publishing translations

u can publish the language files for translations (if needed) with the following command:

```
ohp artisan vendor:publish --tag=filament-panels-translations
```

ıce this package depends on other Filament packages, you can publish the language files for thos
ckages with the following commands:

```
ohp artisan vendor:publish --tag=filament-actions-translations

ohp artisan vendor:publish --tag=filament-forms-translations

ohp artisan vendor:publish --tag=filament-infolists-translations

ohp artisan vendor:publish --tag=filament-notifications-translations

ohp artisan vendor:publish --tag=filament-tables-translations

ohp artisan vendor:publish --tag=filament-translations
```

Upgrading

Upgrading from Filament v2? Please review the upgrade guide.

ament automatically upgrades to the latest non-breaking version when you run `composer update`
ter any updates, all Laravel caches need to be cleared, and frontend assets need to be republishe
u can do this all at once using the `filament:upgrade` command, which should have been added to
ur `composer.json` file when you ran `filament:install` the first time:

```
'post-autoload-dump": [
    // ...
    "@php artisan filament:upgrade"
],
```

ease note that `filament:upgrade` does not actually handle the update process, as Composer doe
ıt already. If you're upgrading manually without a `post-autoload-dump` hook, you can run the
mmand yourself:

```
composer update
```

```
php artisan filament:upgrade
```

it on GitHub

ill need help? Join our Discord community or open a GitHub discussion

Ne

Getting started

Looking for a job using Filament?

Your logo here? 💖