

Improving developer experience in Open Source Projects

How to ease contribution and management of your next big
idea



Hi! I'm Tommaso Bailetti



@Tbaile

Who is Nethesis?

We're an open source company, find us on GitHub!



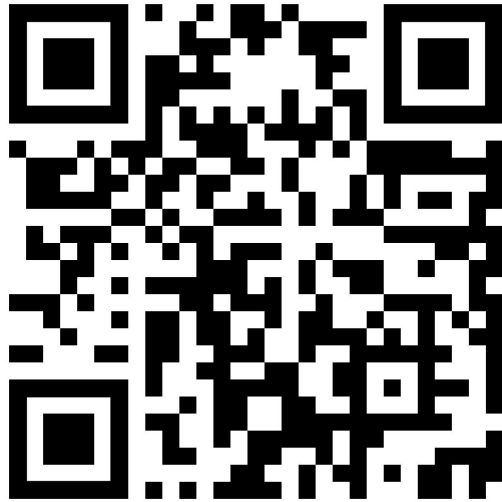
<https://nethserver.org>



<https://nethesis.it>



We even have an international community!



<https://community.nethserver.org>



Do we receive meaningful contributions?

No*



Easy to say that, but how hard it is to contribute?



Nobody knew for sure, so I've done some digging



Everything is blurry, and not connected

Some repos don't have even READMEs

“What is this project for, again?”

“You wanna know why X? Browse the code”



Why is it so hard to write documentation?

You need to write it.



Why is it so hard to write documentation?

However, a nice helper came recently to the rescue: AI!*



*please check the output, hallucinations are around the corner.



Why do people ask always the same questions?



Why do people ask always the same questions?

You are probably the problem
of your documentation.

Sadly, it's a fact!



Why do people ask always the same questions?

You know the software, others don't. You will let out something that others don't know about at all.

Try to let others inexperienced people write it or get feedback and questions from them, will help the onboarding and general feeling of the project by a lot.



Honorable mention: outdated docs

*“I DO AS DOC SAYS, BUT IT
DOESN'T WORK”*



Honorable mention: outdated docs

This is one of the most infuriating one, be wary of this issue!

People will most likely drop the project without even telling you that they found the issue.



Outdated docs: best case scenario



What can reduce our time to code?

Virtualization!



From zero to hero, time is important

Does your project require external deps?

Is yes, how many? Which tool is used to install them?

Do you support every OS/Arch?

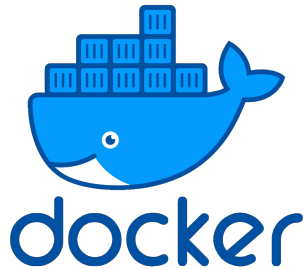
This is too much time consuming.



What can reduce our time? Containers!

Containers are one of the most popular development environments out there.

They remove mostly of the prerequisites to run your project, even helps you mimic a full production environment!



Containers are small for me, what are the alternatives?

There are cases where you need a full system to run testing (systemd, packet manager, and so on...).

Build your own ISO (just for PROs)

Use a tool like Vagrant!



What else?

The *unforeseeable* job: Processes



Processes...?

This last step is probably the most important one.

Defining what code style your project has, what linting, how to contribute or open an issue and even how to handle releases is crucial to allow others to understand what to do and when.



checkstyle 



Processes!

Once you've set all, you can even set back and allow some CI/CD to check this work for you, without even telling the contributors that something is wrong.

Less communication is the most effective communication!



Case of study: Laravel



Forge

Vapor

Ecosystem ▾

News

Partners

Shop



DOCUMENTATION



The PHP Framework for Web Artisans



Laravel is a web application framework with expressive, elegant syntax. We've already laid the foundation — freeing you to create without sweating the small things.



GET STARTED

WATCH LARACASTS



Tbaile @ SFSCON 2023

Case of study: Laravel

Introduction

Instead of defining all of your request handling logic as closures in your route files, you may wish to organize this behavior using "controller" classes. Controllers can group related request handling logic into a single class. For example, a `UserController` class might handle all incoming requests related to users, including showing, creating, updating, and deleting users. By default, controllers are stored in the `app/Http/Controllers` directory.

Writing Controllers

Basic Controllers

To quickly generate a new controller, you may run the `make:controller` Artisan command. By default, all of the controllers for your application are stored in the `app/Http/Controllers` directory:

```
php artisan make:controller UserController
```

Let's take a look at an example of a basic controller. A controller may have any number of public methods which will respond to incoming HTTP requests:

```
<?php

namespace App\Http\Controllers;

use App\Models\User;
use Illuminate\View\View;

class UserController extends Controller
{
    /**
     * Show the profile for a given user.
     */
    public function show(string $id): View
    {
        return view('user.profile', [
            'user' => User::findOrFail($id)
        ]);
    }
}
```



Case of study: Laravel

Laravel & Docker

We want it to be as easy as possible to get started with Laravel regardless of your preferred operating system. So, there are a variety of options for developing and running a Laravel project on your local machine. While you may wish to explore these options at a later time, Laravel provides [Sail](#), a built-in solution for running your Laravel project using [Docker](#).

Docker is a tool for running applications and services in small, light-weight "containers" which do not interfere with your local machine's installed software or configuration. This means you don't have to worry about configuring or setting up complicated development tools such as web servers and databases on your local machine. To get started, you only need to install [Docker Desktop](#).

Laravel Sail is a light-weight command-line interface for interacting with Laravel's default Docker configuration. Sail provides a great starting point for building a Laravel application using PHP, MySQL, and Redis without requiring prior Docker experience.

Laravel Homestead

Introduction

Installation & Setup

- # First Steps

- # Configuring Homestead

- # Configuring Nginx Sites

- # Configuring Services

- # Launching The Vagrant Box

- # Per Project Installation

- # Installing Optional Features

- # Aliases

Updating Homestead

Daily Usage

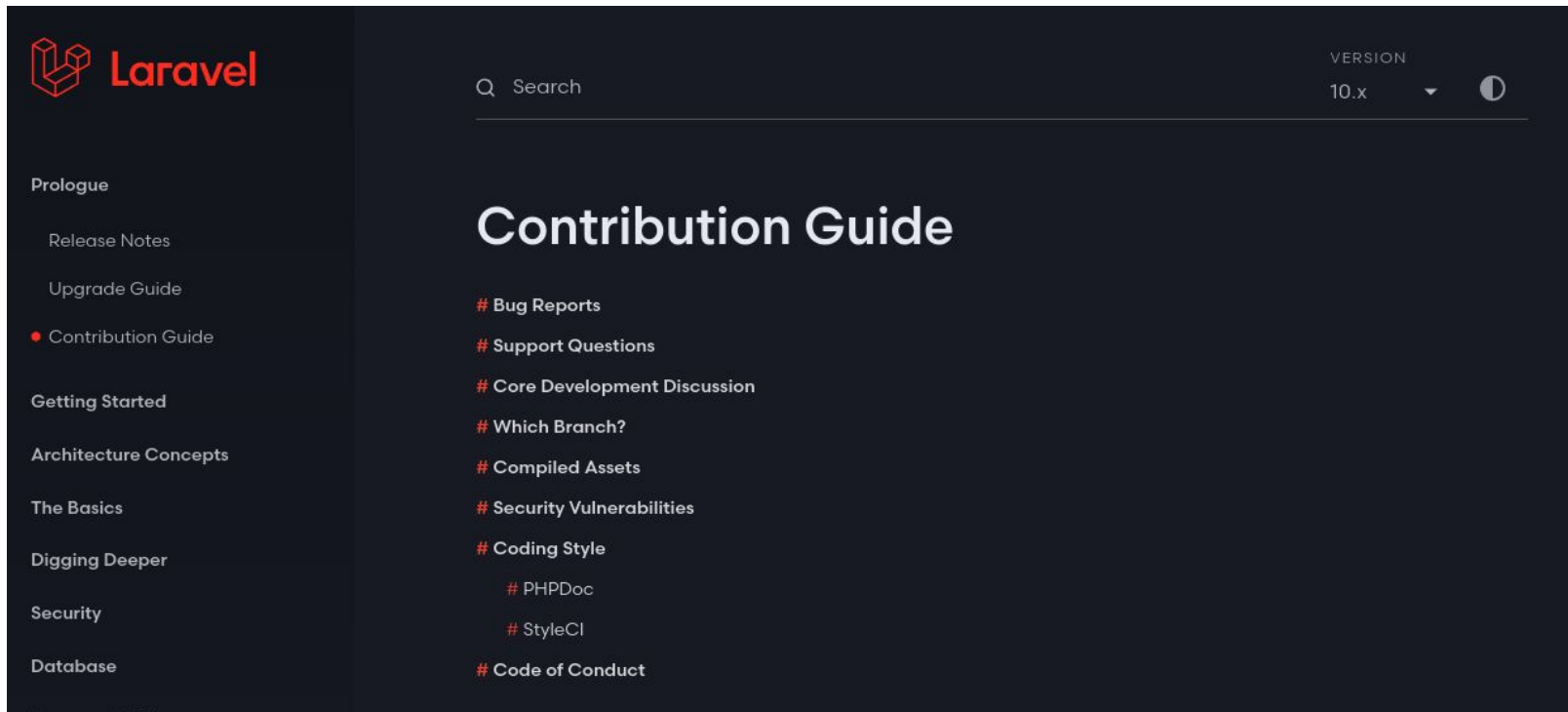
- # Connecting Via SSH

- # Adding Additional Sites

- # Environment Variables



Case of study: Laravel



The screenshot shows the Laravel website's Contribution Guide page. The top left features the Laravel logo. A search bar is located at the top center. In the top right, there is a 'VERSION' dropdown menu set to '10.x' and a moon icon for dark mode. The left sidebar contains a navigation menu with the following items: Prologue, Release Notes, Upgrade Guide, Contribution Guide (highlighted with a red dot), Getting Started, Architecture Concepts, The Basics, Digging Deeper, Security, and Database. The main content area is titled 'Contribution Guide' and lists several topics with red hash symbols: # Bug Reports, # Support Questions, # Core Development Discussion, # Which Branch?, # Compiled Assets, # Security Vulnerabilities, # Coding Style (with sub-items # PHPDoc and # StyleCI), and # Code of Conduct.



Thank you for your attention!

See you next time and have a nice evening!

