

Prologue

Getting Started

Installation

Configuration

Directory Structure

Homestead

Valet

• Deployment

Architecture Concepts

The Basics

Frontend

Security

Digging Deeper

Database

Eloquent ORM

Testing

Official Packages



Deploy Laravel and PHP applications on DigitalOcean, Linode, & AWS.

ADS VIA CARBON

Deployment

Introduction

Server Configuration

Nginx

Optimization

Autoloader Optimization

Optimizing Configuration Loading

Optimizing Route Loading

Deploying With Forge

Introduction

When you're ready to deploy your Laravel application to production, there are some important things you can do to make sure your application is running as efficiently as possible. In this document, we'll cover some great starting points for making sure your Laravel application is deployed properly.

Server Configuration

Nginx

If you are deploying your application to a server that is running Nginx, you may use the following configuration file as a starting point for configuring your web server. Most likely, this file will need to be customized depending on your server's configuration. If you would like assistance in managing your server, consider using a service such as [Laravel Forge](#):

```
server {
    listen 80;
    server_name example.com;
    root /example.com/public;

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Content-Type-Options "nosniff";

    index index.html index.htm index.php;

    charset utf-8;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt { access_log off; log_not_found off; }

    error_page 404 /index.php;

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php7.2-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.(!well-known).* {
        deny all;
    }
}
```

Optimization

Autoloader Optimization

When deploying to production, make sure that you are optimizing Composer's class autoloader map so Composer can quickly find the proper file to load for a given class:

```
composer install --optimize-autoloader --no-dev
```



In addition to optimizing the autoloader, you should always be sure to include a `composer.lock` file in your project's source control repository. Your project's dependencies can

be installed much faster when a `composer.lock` file is present.

Optimizing Configuration Loading

When deploying your application to production, you should make sure that you run the `config:cache` Artisan command during your deployment process:

```
php artisan config:cache
```

This command will combine all of Laravel's configuration files into a single, cached file, which greatly reduces the number of trips the framework must make to the filesystem when loading your configuration values.



If you execute the `config:cache` command during your deployment process, you should be sure that you are only calling the `env` function from within your configuration files. Once the configuration has been cached, the `.env` file will not be loaded and all calls to the `env` function will return `null`.

Optimizing Route Loading

If you are building a large application with many routes, you should make sure that you are running the `route:cache` Artisan command during your deployment process:

```
php artisan route:cache
```

This command reduces all of your route registrations into a single method call within a cached file, improving the performance of route registration when registering hundreds of routes.



Since this feature uses PHP serialization, you may only cache the routes for applications that exclusively use controller based routes. PHP is not able to serialize Closures.

Deploying With Forge

If you aren't quite ready to manage your own server configuration or aren't comfortable configuring all of the various services needed to run a robust Laravel application, [Laravel Forge](#) is a wonderful alternative.

Laravel Forge can create servers on various infrastructure providers such as DigitalOcean, Linode, AWS, and more. In addition, Forge installs and manages all of the tools needed to build robust Laravel applications, such as Nginx, MySQL, Redis, Memcached, Beanstalk, and more.

Become a Laravel Partner

Laravel Partners are elite shops providing top-notch Laravel development and consulting. Each of our partners can help you craft a beautiful, well-architected project.

[Our Partners](#)

Laravel

Highlights

[Release Notes](#)
[Getting Started](#)
[Routing](#)
[Blade Templates](#)
[Authentication](#)
[Authorization](#)
[Artisan Console](#)
[Database](#)
[Eloquent ORM](#)
[Testing](#)

Resources

[Laracasts](#)
[Laravel News](#)
[Laracon](#)
[Laracon EU](#)
[Laracon AU](#)
[Jobs](#)
[Certification](#)
[Forums](#)

Partners

[Vehikl](#)
[Tighten Co.](#)
[Kirschbaum](#)
[Byte 5](#)
[64Robots](#)
[Cubet](#)
[DevSquad](#)
[Ideil](#)
[Cyber-Duck](#)
[ABOUT YOU](#)
[Become A Partner](#)

Ecosystem

[Vapor](#)
[Forge](#)
[Envoyer](#)
[Horizon](#)
[Lumen](#)
[Nova](#)
[Echo](#)
[Valet](#)
[Mix](#)
[Spark](#)
[Cashier](#)
[Homestead](#)
[Dusk](#)
[Passport](#)
[Scout](#)
[Socialite](#)

Laravel is a web application framework with expressive, elegant syntax. We believe development must be an enjoyable and creative experience to be truly fulfilling. Laravel attempts to take the pain out of development by easing common tasks used in most web projects.

Laravel is a Trademark of Taylor Otwell.
Copyright © 2011-2019 Laravel LLC.

