
Perform Self-Diagnosis Tests On Your Laravel Application

downloads 1.1M

downloads 1.1M

This package allows you to run self-diagnosis tests on your Laravel application. It comes with multiple checks out of the box and allows you to add custom checks yourself.

Here is an example output of the command:

```
➤ ~/Code/self-diagnosis-test ➤ master ➤ php artisan self-diagnosis
|-----|
| Running Common Checks
|-----|
Running check 1/10: App key is set... ✓
Running check 2/10: Composer dependencies are up to date... ✓
Running check 3/10: The correct PHP version is installed... ✓
Running check 4/10: The database can be accessed... ✓
Running check 5/10: The migrations are up to date... ✓
Running check 6/10: The required PHP extensions are installed... ✓
Running check 7/10: The environment file exists... ✓
Running check 8/10: The example environment variables are set... ✓
Running check 9/10: The directories have the correct permissions.... ✓
Running check 10/10: The storage directory is linked.... ✓

|-----|
| Environment Specific Checks (local)
|-----|
Running check 1/2: Configuration is not cached... ✓
Running check 2/2: Routes are not cached... ✓

Good job, looks like you are all set up.
```

Included checks

- Is the APP_KEY set?
- Are your composer dependencies up to date with the composer.lock file?
- Do you have the correct PHP version installed?
- Do you have the correct PHP extensions installed?
- Can a connection to the database be established?
- Do the `storage` and `bootstrap/cache` directories have the correct permissions?
- Does the `.env` file exist?
- Is the maintenance mode disabled?
- Are the required locales installed on the system?

-
- Are there environment variables that exist in `.env.example` but not in `.env`?
 - Are there any migrations that need to be run?
 - Is the storage directory linked?
 - Can Redis be accessed?

Development environment checks

- Is the configuration not cached?
- Are the routes not cached?
- Are there environment variables that exist in `.env` but not in `.env.example`?

Production environment checks

- Is the configuration cached?
- Are the routes cached?
- Is the xdebug PHP extension disabled?
- Is APP_DEBUG set to false?
- Are certain servers reachable?
- Are certain supervisor programs running?

Installation

You can install the package via composer:

```
1 composer require beyondcode/laravel-self-diagnosis
```

Usage

Just call the artisan command to start the checks:

```
1 php artisan self-diagnosis
```

Customization

You can publish the configuration file, that contains all available checks using:

```
1 php artisan vendor:publish --provider=BeyondCode\\SelfDiagnosis\\  
  SelfDiagnosisServiceProvider
```

This will publish a `self-diagnosis.php` file in your `config` folder. This file contains all the checks that will be performed on your application.

```
1 <?php
2
3 return [
4
5     /*
6      * A list of environment aliases mapped to the actual environment
7      * configuration.
8      */
9     'environment_aliases' => [
10         'prod' => 'production',
11         'live' => 'production',
12         'local' => 'development',
13     ],
14
15     /*
16      * Common checks that will be performed on all environments.
17      */
18     'checks' => [
19         \BeyondCode\SelfDiagnosis\Checks\AppKeyIsSet::class,
20         \BeyondCode\SelfDiagnosis\Checks\CorrectPhpVersionIsInstalled::
21             class,
22         \BeyondCode\SelfDiagnosis\Checks\DatabaseCanBeAccessed::class
23         => [
24             'default_connection' => true,
25             'connections' => [],
26         ],
27         \BeyondCode\SelfDiagnosis\Checks\
28             DirectoriesHaveCorrectPermissions::class => [
29             'directories' => [
30                 storage_path(),
31                 base_path('bootstrap/cache'),
32             ],
33         ],
34         \BeyondCode\SelfDiagnosis\Checks\EnvFileExists::class,
35         \BeyondCode\SelfDiagnosis\Checks\
36             ExampleEnvironmentVariablesAreSet::class,
37         \BeyondCode\SelfDiagnosis\Checks\LocalesAreInstalled::class =>
38         [
39             'required_locales' => [
40                 'en_US',
41                 'en_US.utf8',
42             ],
43         ],
44         \BeyondCode\SelfDiagnosis\Checks\MaintenanceModeNotEnabled::
45             class,
46         \BeyondCode\SelfDiagnosis\Checks\MigrationsAreUpToDate::class,
47         \BeyondCode\SelfDiagnosis\Checks\PhpExtensionsAreInstalled::
48             class => [
```

```

41         'extensions' => [
42             'openssl',
43             'PDO',
44             'mbstring',
45             'tokenizer',
46             'xml',
47             'ctype',
48             'json',
49         ],
50         'include_composer_extensions' => true,
51     ],
52     \BeyondCode\SelfDiagnosis\Checks\StorageDirectoryIsLinked::
53         class,
54 ],
55 /*
56  * Environment specific checks that will only be performed for the
57  * corresponding environment.
58  */
59 'environment_checks' => [
60     'development' => [
61         \BeyondCode\SelfDiagnosis\Checks\
62             ComposerWithDevDependenciesIsUpToDate::class => [
63                 'additional_options' => '--ignore-platform-reqs',
64             ],
65         \BeyondCode\SelfDiagnosis\Checks\ConfigurationIsNotCached::
66             class,
67         \BeyondCode\SelfDiagnosis\Checks\RoutesAreNotCached::class,
68     ],
69     'production' => [
70         \BeyondCode\SelfDiagnosis\Checks\
71             ComposerWithoutDevDependenciesIsUpToDate::class => [
72                 'additional_options' => '--ignore-platform-reqs',
73             ],
74         \BeyondCode\SelfDiagnosis\Checks\ConfigurationIsCached::
75             class,
76         \BeyondCode\SelfDiagnosis\Checks\DebugModeIsNotEnabled::
77             class,
78         \BeyondCode\SelfDiagnosis\Checks\PhpExtensionsAreDisabled::
79             class => [
80                 'extensions' => [
81                     'xdebug',
82                 ],
83             ],
84         \BeyondCode\SelfDiagnosis\Checks\RedisCanBeAccessed::class
85             => [
86             'default_connection' => true,
87             'connections' => [],
88         ],
89         \BeyondCode\SelfDiagnosis\Checks\RoutesAreCached::class,
90         \BeyondCode\SelfDiagnosis\Checks\ServersArePingable::class

```

```

83         => [
84             'servers' => [
85                 'www.google.com',
86                 ['host' => 'www.google.com', 'port' => 8080],
87                 ['host' => '8.8.8.8', 'port' => 8080, 'timeout' =>
                    5],
88             ],
89         ],
90         \BeyondCode\SelfDiagnosis\Checks\
          SupervisorProgramsAreRunning::class => [
91             'programs' => [
92                 'horizon',
93             ],
94             'restarted_within' => 300, // max seconds since last
                                     restart, 0 to disable check
95         ],
96     ],
97 ],
98
99 ];

```

Available Configuration Options The following options are available for the individual checks:

- `BeyondCode\SelfDiagnosis\Checks\ComposerWithDevDependenciesIsUpToDate`
 - **additional_options** (string, optional parameters like `'--ignore-platform-reqs'`): optional arguments for composer
- `BeyondCode\SelfDiagnosis\Checks\ComposerWithoutDevDependenciesIsUpToDate`
 - **additional_options** (string, optional parameters like `'--ignore-platform-reqs'`): optional arguments for composer
- `BeyondCode\SelfDiagnosis\Checks\DatabaseCanBeAccessed`
 - **default_connection** (boolean, default: **true**): if the default connection should be checked
 - **connections** (array, list of connection names like `['mysql', 'sqlsrv']`, default: `[]`): additional connections to check
- `BeyondCode\SelfDiagnosis\Checks\DirectoriesHaveCorrectPermissions`
 - **directories** (array, list of directory paths like `[storage_path(), base_path('bootstrap/cache')]`, default: `[]`): directories to check

-
- `BeyondCode\SelfDiagnosis\Checks\LocalesAreInstalled`
 - **required_locales** (array, list of locales like [`'en_US'`, `'en_US.utf8'`], default: []): locales to check
 - `BeyondCode\SelfDiagnosis\Checks\PhpExtensionsAreDisabled`
 - **extensions** (array, list of extension names like [`'xdebug'`, `'zlib'`], default: []): extensions to check
 - `BeyondCode\SelfDiagnosis\Checks\PhpExtensionsAreInstalled`
 - **extensions** (array, list of extension names like [`'openssl'`, `'PDO'`], default: []): extensions to check
 - **include_composer_extensions** (boolean, default: **false**): if required extensions defined in `composer.json` should be checked
 - `BeyondCode\SelfDiagnosis\Checks\RedisCanBeAccessed`
 - **default_connection** (boolean, default: **true**): if the default connection should be checked
 - **connections** (array, list of connection names like [`'cache_1'`, `'cache_2'`], default: []): additional connections to check
 - `BeyondCode\SelfDiagnosis\Checks\SupervisorProgramsAreRunning`
 - **programs** (array, list of programs like [`'horizon'`, `'another-program'`], default: []): programs that are required to be running
 - **restarted_within** (integer, max allowed seconds since last restart of programs (0 = disabled), default: 0): verifies that programs have been restarted recently to grab code updates
 - `BeyondCode\SelfDiagnosis\Checks\ServersArePingable`
 - **servers** (array, list of servers and parameters like [`'google.com'`, [`'host'=> 'google.com'`, `'port'=> 8080`]]): servers to ping

Custom Checks

You can create custom checks, by implementing the `BeyondCode\SelfDiagnosis\Checks\Check` interface and adding the class to the config file. Like this:

```
1 <?php
2
3 use BeyondCode\SelfDiagnosis\Checks\Check;
```

```
4
5 class MyCustomCheck implements Check
6 {
7     /**
8      * The name of the check.
9      *
10     * @param array $config
11     * @return string
12     */
13     public function name(array $config): string
14     {
15         return 'My custom check.';
16     }
17
18     /**
19     * Perform the actual verification of this check.
20     *
21     * @param array $config
22     * @return bool
23     */
24     public function check(array $config): bool
25     {
26         return true;
27     }
28
29     /**
30     * The error message to display in case the check does not pass.
31     *
32     * @param array $config
33     * @return string
34     */
35     public function message(array $config): string
36     {
37         return 'This is the error message that users see if "check"
38             returns false.';
39     }
40 }
```

Example Output

```
↑ ~/Code/self-diagnosis-test > master$ php artisan self-diagnosis
|-----|
| Running Common Checks
|-----|
Running check 1/10: App key is set... ✓
Running check 2/10: Composer dependencies are up to date... ✓
Running check 3/10: The correct PHP version is installed... ✓
Running check 4/10: The database can be accessed... ✓
Running check 5/10: The migrations are up to date... ✗
Running check 6/10: The required PHP extensions are installed... ✓
Running check 7/10: The environment file exists... ✓
Running check 8/10: The example environment variables are set... ✗
Running check 9/10: The directories have the correct permissions.... ✓
Running check 10/10: The storage directory is linked.... ✓

|-----|
| Environment Specific Checks (production)
|-----|
Running check 1/4: Configuration is cached... ✗
Running check 2/4: Routes are cached... ✗
Running check 3/4: The xdebug extension is not active.... ✓
Running check 4/4: Debug mode is not enabled... ✗

The following checks failed:
You need to update your migrations. Call "php artisan migrate".

These example environment variables are missing in your .env file, but are defined in your .env.example:
SLACK_KEY
SLACK_SECRET
SLACK_REDIRECT_URI

Your configuration files should be cached in production. Call "php artisan config:cache" to cache the configuration.

Your routes should be cached in production. Call "php artisan route:cache" to create the route cache.

You should not use debug mode in production. Set APP_DEBUG to false.
```

Testing

```
1 composer test
```

Changelog

Please see CHANGELOG for more information what has changed recently.

Contributing

Please see CONTRIBUTING for details.

Security

If you discover any security related issues, please email marcel@beyondco.de instead of using the issue tracker.

Credits

- Marcel Pociot
- All Contributors

License

The MIT License (MIT). Please see License File for more information.