Log automation - software with php laravel

If you are a good developer, you might be always having a keen attention on the potential errors while developing a web application. You might be setting up workflows in a way to help you log errors properly. With PHP, you get some of the most robust method to manage the PHP error logs which can be logged manually and then automated. You will also find some third party tools that are developed by the open source community for handling the PHP error logging process.

What you should think upon is that when the errors should be logged and how that should be done? While working in the dev mode, you can log PHP errors as per your preference. You can create a PHP error log file or save them as notifications on different channels. You can always log errors conveniently as per your requirements.

### Getting Started with PHP Error Logging

PHP is the most popular and widely used programming language for developing web applications. As per the builtwith insights, today almost 37,805,937 websites use PHP as their backend language, which makes it around 75% of all the websites in the world. These stats clearly display that PHP still has a highest market share in the programming world.

While developing an application in PHP, you use few commands like print\_r(),var\_dump() to debug errors and log on browser. But while working in the production mode, that is not the safest way. In dev mode, you can do it, but you need to disable it when initiating the migration. Therefore, it is easy to log errors in PHP with error\_log() function in dev mode, which sends an error message to the defined error handling routines.

Let’s imagine that you are connecting MySQL database with PHP and if it fails to connect with it, you can log errors in PHP as below:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <?php  // Send error message to the server log if error connecting to the database  if (!mysqli\_connect("localhost","bad\_user","bad\_password","my\_db")) {     error\_log("Failed to connect to database!", 0);  }  // Send email to administrator if we run out of FOO  if (!($foo = allocate\_new\_foo())) {     error\_log("Oh no! We are out of FOOs!", 1, "admin@example.com");  }  ?> |

### Enabling Error Logging in php.ini

In order to log errors in PHP, open the php.ini file and uncomment/add the below lines of code:



|  |  |
| --- | --- |
| 1  2  3  4 | error\_reporting = E\_ALL & ~E\_NOTICE  error\_reporting = E\_ALL & ~E\_NOTICE | E\_STRICT  error\_reporting = E\_COMPILE\_ERROR|E\_RECOVERABLE\_ERROR|E\_ER… \_ERROR  error\_reporting = E\_ALL & ~E\_NOTICE |

For enabling the error logging in individual files, add this code at the top of the PHP file.



|  |  |
| --- | --- |
| 1  2  3 | ini\_set('display\_errors', 1);  ini\_set('display\_startup\_errors', 1);  error\_reporting(E\_ALL); |

Now, you should enable only one statement to parse the log errors in php.ini file:



|  |  |
| --- | --- |
| 1 | display\_errors = on. |

With this you can easily see logged errors in your browser. Some of the additional commands you can write for PHP error logging include:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | // Turn off all error reporting  error\_reporting(0);    // Report simple running errors  error\_reporting(E\_ERROR | E\_WARNING | E\_PARSE);    // Reporting E\_NOTICE can be good too (to report uninitialized variables or catch variable name misspellings ...)  error\_reporting(E\_ERROR | E\_WARNING | E\_PARSE | E\_NOTICE);    // Report all errors except E\_NOTICE  error\_reporting(E\_ALL & ~E\_NOTICE);    // Report all PHP errors (see changelog)  error\_reporting(E\_ALL);    // Report all PHP errors  error\_reporting(-1);    // Same as error\_reporting(E\_ALL);  ini\_set('error\_reporting', E\_ALL); |

### Copying PHP Error Logs to File

The above mentioned practices work well when you work in the dev environment. But, while taking your website live and starting your work in the production mode, it is essential to hide the errors from on screen display and log them within a backend file. These can be saved at a specific PHP error logs location whose URL is already described in the php.ini file.

PHP stores error logs in**/var/log/apache2**, if it uses an apache2 module. Shared hosts have a tendency to store PHP error log files in your root directory **/log subfolder**. But, if you have access to the php.ini file you can do this as below:



|  |  |
| --- | --- |
| 1 | error\_log = /var/log/php-scripts.log |

In case you are using cPanel, the master log file, the thing you’re looking for is stored (by default) at:



|  |  |
| --- | --- |
| 1 | /usr/local/apache/logs/error\_log |

If everything fails, you can check the PHP error logs location with the below code:



|  |  |
| --- | --- |
| 1 | <?php phpinfo(); ?> |

### Error Logging in PHP Frameworks

The above steps are for core PHP development. PHP has several Model View Architecture (MVC) and micro-frameworks which have their own methods on top of the core functions. Some of the best PHP MVC frameworks to build advanced web applications with comprise of: Laravel, CodeIgniter and Symfony.

### Error Logging In Laravel

After setting up a new Laravel project, error logging and exception handling come pre-configured in it. Laravel has a separate class **App\Exceptions\Handler** that handles all these issues. For logging Laravel errors, it uses the Monolog library that supports array of log handlers. Several of these handlers are configured by Laravel for you, letting you to choose between a single PHP error log file, rotating log files and writing error information to the system log.

You can configure the PHP **debug** option in **config/app.php** file to log the errors on the display of user. This option’s value can be set in .env file under property **APP\_DEBUG** which is an environment variable, thus setting up in **app.php** file. When the development is done locally, the value can be set to **true**, and after migrating to production, it should be set as **false**. Otherwise, the security concern always remains there, as it shows error on browser screen.

Being the developer, you can save Laravel log information on a single file, daily file, syslog and the errorlog. For configuring these options for Laravel error logging, open app.php file and edit the log option. For example, if you want to setup daily logging for errors, follow the below code:



|  |  |
| --- | --- |
| 1 | 'log' => 'daily' |

Monolog is enable to log errors with different security warnings. It adds all the errors in storage by default, but you can identify them as error, alert, emergency, critical and warning. For this, add **‘log\_level’** property in options as below:



|  |  |
| --- | --- |
| 1 | 'log\_level' => env('APP\_LOG\_LEVEL', 'error'), |

### Automating PHP Error Logging Process

The complete work of error logging can be done manually, but while working in multiple teams and larger projects, you must setup an automated workflow which can log errors in third party services like Slack, Sentry, Blackfire etc.

With these tools you can get a better understanding of errors and their solutions. It is possible to setup Slack channels to send quick notifications to the teams about any runtime incidents.

Suppose that you have already setup Slack channel and webhooks as below:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | monolog:     handlers:         main:             type:  fingers\_crossed             action\_level: error             handler:  nested         nested:             type: stream             path: "%kernel.logs\_dir%/%kernel.environment%.log"             level: debug         console:             type: console         slack:             type: slack             token: xxxx-xxxxxxxxxxx-xxxxxxxxx-xxxxxxxxxx-xxxxxx             channel: "#name-of-channel"             bot\_name: ChooseName             icon\_emoji: :ghost:             level: critical |

In the above example, you will also find the emojis setup? This indicates the ways you can send Slack notifications to any particular channel as per your needs.