**How to Clear "Exception Handler Doesn't Show Actual File & Line of Exception" in Magento 2**  
  
**1. Enable Developer Mode in Magento 2**

Ensure that your Magento 2 store is in **Developer Mode**, which will provide detailed error reporting, including file and line numbers.

* To enable Developer Mode:

php bin/magento deploy:mode:set developer

This will ensure that Magento displays detailed stack traces, which includes the file and line number where the error occurs.

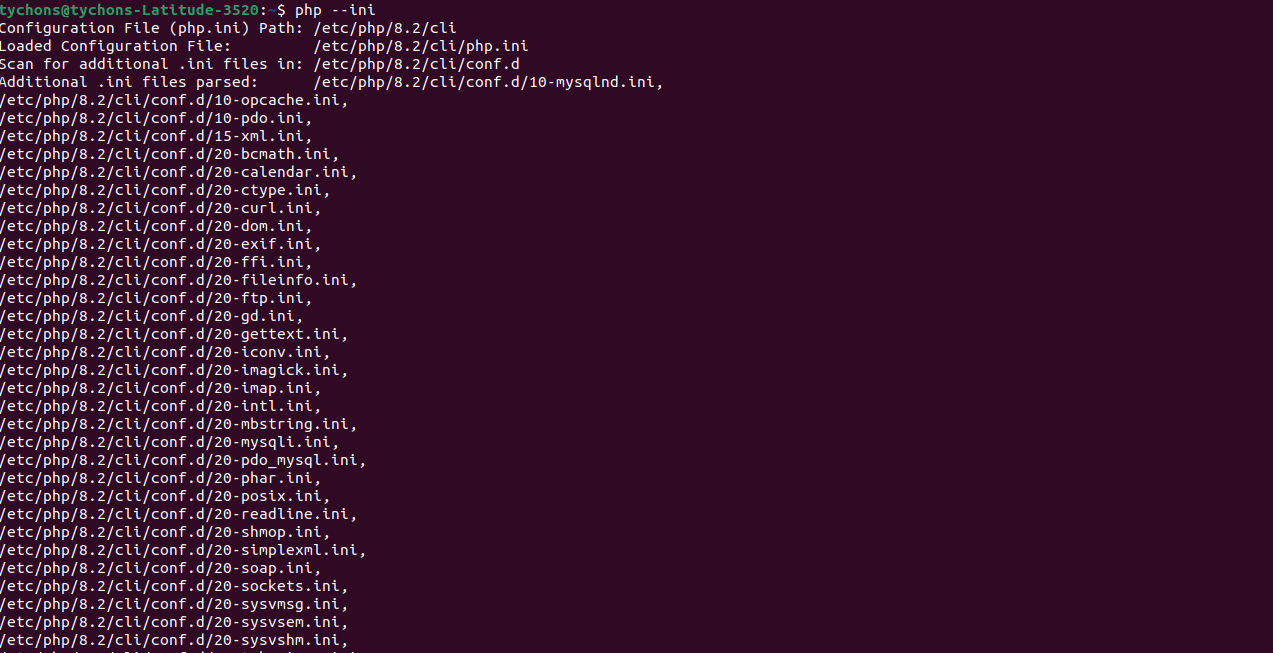
**2. Check php.ini Settings**

Magento 2 relies on PHP error logging to capture detailed exception information. Ensure that PHP error logging is enabled in the php.ini file.

* Open your php.ini file and ensure the following settings are correctly set:

display\_errors = On  
 error\_reporting = E\_ALL  
 log\_errors = On  
 error\_log = /path/to/your/php-error.log

* After making changes to php.ini, restart your web server to apply the new settings.



**3. Enable Exception Logging in Magento 2**

Magento 2 has a built-in logging system that records exceptions. Ensure that exception logging is enabled in the system.

* Open the app/bootstrap.php file and ensure that exception logging is enabled:

php

if ($bootstrap->getEnvironment() == \Magento\Framework\App\State::MODE\_DEVELOPER) {  
 $logger = $objectManager->get(\Psr\Log\LoggerInterface::class);  
 $logger->addHandler(new \Monolog\Handler\StreamHandler(BP . '/var/log/system.log'));  
}

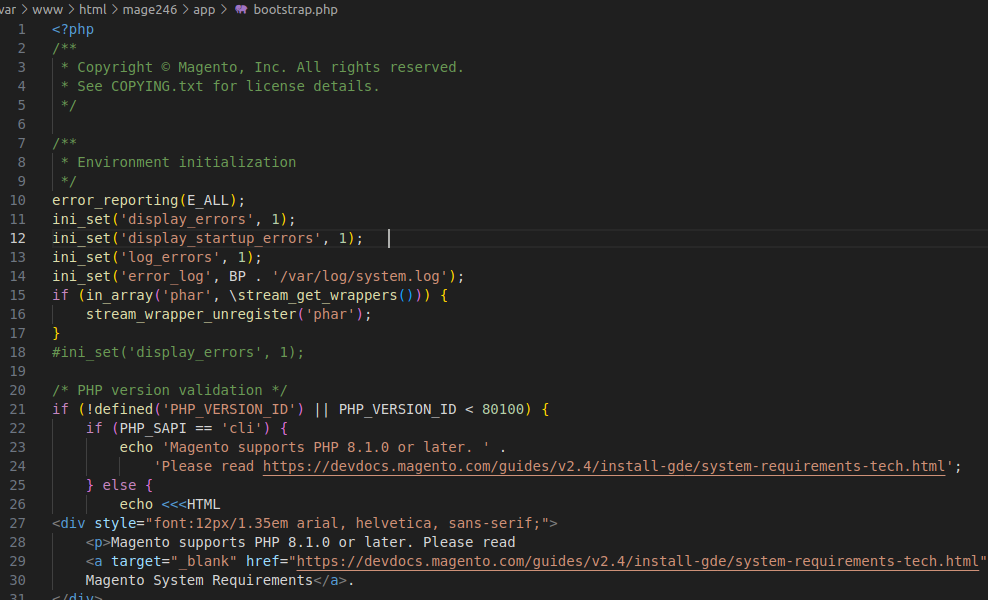
**4. Use Xdebug for Debugging**

For a more granular debugging experience, you can install and configure **Xdebug**. Xdebug will allow you to step through the code and view the exact file and line where exceptions are thrown.

* Install Xdebug by following the instructions for your server's environment (Apache, Nginx, etc.).
* Once installed, enable it in your php.ini:

zend\_extension=xdebug.so  
xdebug.remote\_enable = 1  
xdebug.remote\_host = 127.0.0.1  
xdebug.remote\_port = 9000  
xdebug.remote\_autostart = 1

* After configuring Xdebug, use an IDE (like PhpStorm) to debug your Magento application. Set breakpoints and analyze the stack trace in more detail.



**5. Update Magento and PHP to the Latest Version**

Sometimes, the issue may be related to a bug that has been fixed in a later release. Make sure you are using the latest version of Magento 2 and PHP.

* To update Magento, run the following commands:

composer update  
php bin/magento setup:upgrade  
php bin/magento cache:flush

* Ensure your PHP version is compatible with the Magento version you're using. Magento 2.4.x requires at least PHP 7.4.

**6. Modify Error Handling in index.php (if needed)**

If the error still doesn’t show the correct file and line number, you can modify the error handling in index.php (found in the root of the Magento directory) to ensure exceptions are logged correctly.

* Open index.php and ensure that the following is present to catch uncaught exceptions and display detailed information:

set\_error\_handler(function ($severity, $message, $file, $line) {  
 if ($severity & error\_reporting()) {  
 echo "Error: [$severity] $message in $file on line $line";  
 }  
});

* This will display errors with file names and line numbers for any uncaught exceptions.

**7. Check Magento Logs in var/log/**

After enabling detailed logging and debugging, check Magento's var/log/ directory for error logs.

* The following files may provide useful information:
  + exception.log
  + system.log

You can find these logs in var/log/:

tail -f var/log/exception.log  
tail -f var/log/system.log

These logs should contain information about the exceptions, including the file name and line number.

**8. Clear Cache and Recompile**

After making these changes, clear Magento's cache and recompile the application to ensure the changes take effect.

* Run the following commands:

bash

Copy code

php bin/magento cache:flush  
php bin/magento setup:di:compile  
php bin/magento setup:static-content:deploy

**9. Check for Custom Modules or Extensions**

If the issue is specific to custom modules or third-party extensions, temporarily disable them to isolate the cause. You can disable extensions by running:

php bin/magento module:disable Vendor\_ModuleName  
php bin/magento cache:flush

After identifying the problematic module, review its exception handling and logs.

**Summary**

To resolve the "Exception Handler doesn't show actual file & line of exception thrown" issue in Magento 2, you need to:

1. Enable Developer Mode.
2. Ensure PHP error reporting is set correctly in php.ini.
3. Enable exception logging in Magento 2.
4. Use Xdebug for detailed debugging.
5. Ensure Magento and PHP are updated to the latest versions.
6. Review error handling and logging configurations in index.php.

By following these steps, you should be able to identify and display the actual file and line number for any exceptions thrown in Magento 2.

---------------------------------------------------------------------