

A very lightweight logging module that supports

- Rest Api Logging
- Elastic Search Logging
- GraphQL Logging
- RabbitMq Logging
- Varnish Cache Tag Bans Logging

No Core files is overridden or modified. These logging are implemented using plugins.

## To Install

**For Magento Version >= 2.4.4**

```
composer require moises/magento-log  
php bin/magento setup:upgrade
```

**For Magento Version < 2.4.4**

```
composer require moises/magento-log 1.1.0  
php bin/magento setup:upgrade
```

### Supported Logging

- Rest API Logging
- ElasticSearch Logging

For Magento version less than 2.4.4 if you need support for GraphQL logging plz try the below (Note its not tested from my end you can give a try)

- Add this commit in to your code  
base <https://github.com/magento/magento2/commit/50c88eb432b4a0880484a4db46165ad431d12a14>

- Post that download this [module](#) version 2.1.0 directly from git and then put it under the app/code

Warning : This package cannot be installed via Composer 1.x, please make sure you upgrade to Composer 2+.

Alternatively you can download from github as well <https://github.com/mosesdinakaran/magento-log.git>

### apilogging

```

1 2 3 4
[{"REQUEST":{"request_url":"/rest/V1/catalog/product","path_info":"/rest/V1/catalog/product","body_params":{"cart_item":{"quote_id":"503942","product_type":"configurable","sku":"0568597006","qty":1,"product_option":{"extension_attributes":{"configurable_item_options":{"option_id":"383","option_value":"043"},{"option_id":"865","option_value":"39940"}}}}},"params":{"XDEBUG_SESSION_START":"PHPSTORM"},"http_method":"POST","client_ip":"172.17.32.1","headers":{"Cookie":"PHPSESSID=mu16an2q7afv1216a040qk16; XDEBUG_SESSION=PHPSTORM; private_content_version=67fc131fc6e94a1467d3bed8b6afic","Content-Length":"344","Host":"hmn.development","Postman-Token":"26ab2df9-2837-45e8-a4f8-074db1b28b4","Content-Type":"application/json","Authorization":"Bearer gvs5ddk8h8qp7871a31zmmagmddq72","version":"1.1","schema":"http"},"RESPONSE":{"headers":{"X-Frame-Options":"SAMEORIGIN","Content-Type":"application/json; charset=utf-8"},"status_code":200,"version":"1.1","exception":[],"response_body":{"item_id":"397653","sku":"RM0568597006202001","qty":1,"name":"Buit trousers","price":10.5,"product_type":"configurable","quote_id":"503942","product_option":{"extension_attributes":{"configurable_item_options":{"option_id":"383","option_value":"043"},{"option_id":"865","option_value":"39940"}}}}]}]}

```

Author: Moses Dinakaran

## Rest API Logging

This API logging extension provides an options to log all/specific api calls.

In most of our environment we use different third party services such as order management, product management, product search , shipping management etc. These services interact with Magento using Rest API'S.

At times it would become very difficult to identify the root cause if we face any problem with these services.

Moreover if we are in headless environment primarily using rest apis it would be really hard to identify the root cause for certain issues.

Consider a scenario where add to cart is not working for a particular customer only.

To effectively debug this issue, we would require the post data, headers etc that is posted by that particular user.

This extension would help you in this scenario. It has the feature that we can log the api request that is made by a specific user.

Let see the complete features of this extension.

### **Features**

Enable or Disable Logging through Admin Configuration

Able to configure the api urls that needs to be logged using regular expression which will give endless opportunity to the developer to log a specific url

Few such ex are

- Able to log all the API logs
- Able to log logged in users API calls
- Able to log guest user API calls
- Able to log only a specific api calls such as custom api call, cart, CMS, Product, Order etc
- Able to log a specific user api calls.
- The API logs contain both the request and Response
- The logs are made in a separate file
- This extension doesn't override any CORE Api class.

This can be safely deployed in Production environment as its very light weight.

### **Configuration**

Stores -> Configuration -> Moses Extensions -> API Logging

## API Logging

Enable API Logging  
[website] Enable For Selected Apis

Regular Expression Patterns  
[website] V1/guest-carts./\*/items

- Use New line to enter more than one urls or Regular Expression  
- Ex  
V1/carts/9 : Matches all urls for quote ID 9 that contains V1/carts/9  
V1/carts/(\d)\* : Matches for all quotes  
V1/checkoutcomupapi/getTokenList : To Match a specific url The method preg\_match is used to match the url

Enable API Logging : To Enable all/selected api calls

Regular Expression Patterns: Define the regular expression pattern to log selected urls, For more than one urls use next line.

Ex

V1/carts/9 : Matches all urls for quote ID 9 that contains V1/carts/9

V1/carts/(\d)\* : Matches for all quotes

V1/checkoutcomupapi/getTokenList : Match a specific url

## Output

Once the API Logging is enabled, The request and response will be available in the below log file `MAGE_ROOT/var/log/moses-logging.log`

## Elastic Search Logging

At times there might be a case where you need to find out exactly what data is being pushed to elastic search and what is the response that we receive. This extension will help you to do that. It will log all the request and response of Elasticsearch from Magento.

## Configuration

Stores -> Configuration -> Moses Extensions -> Elastic Search Logging

## **Output**

Once the Elasticsearch Logging is enabled, all the communication between magento and elastic search are logged here `MAGE_ROOT/var/log/moses-logging.log`

## **How it works**

The implementation is not very complicated its quiet simple though.

Magento uses the third party api client "elasticsearch/elasticsearch" to interact with ElasticSearch. This Extension by default has the feature of logging the request and responses.

```
\Elasticsearch\Connections\Connection::logRequestSuccess
```

But while creating the instance of this elasticsearch model, Magento always set the logger to be NULL due to which the the request never get logged. This extension thorough a plugging sets this looger to a custom logger, Due to which the requests and the response are logged.

## **GraphQL Logging**

GraphQL logging is supported only for magento version greater than or equal to 2.4.4

## Graphql Logging

Graphql Logging  
[store view]

No

Filter By Http Headers  
[store view]


Log All Http Methods

Filter By Query Type  
[website]

categories

A comma separated GraphQL query Types  
Ex  
- product : To log only the product query type  
- product,categories : To log both product and categories query types

Filter By Headers  
[website]

Header Key	Header Value	Action
<input type="text"/>	<input type="text"/>	
<div>Add</div>		

## Features

- Able to log GET/POST Requests or All Requests
- Able to log Based on HTTP Header Values
- Able to log Based on Query types
  - product : To log only the product query type
  - product,categories : To log both product and categories query types

## Configuration

Stores -> Configuration -> Moses Extensions -> Graphql Logging

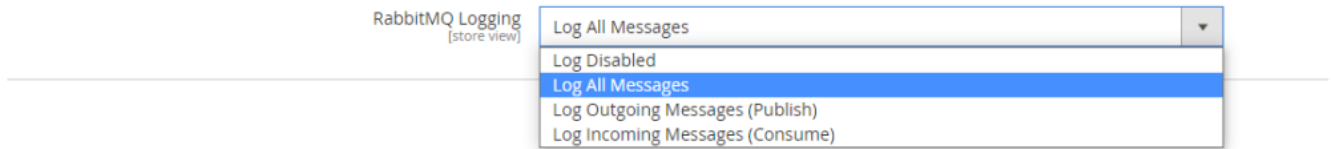
## RabbitMq Logging

To log the RabbitMq Messages that are send from Magento to RabbitMq Server or that are cosumed by Magento from RabbitMq.

## Configuration

Stores -> Configuration -> Moses Extensions -> RabbitMq Logging

## RabbitMQ / MessageQueue Logging



### Features

- Able to log Outgoing Messages (The Messages that are published to RabbitMq Server)
- Able to log Incoming Messages (The messages that are consumed from RabbitMq Server)

## Varnish Cache Tags Logging

To log the varnish purge cache tags, To know more on this [please refer here](#)

### Configuration

Stores -> Configuration -> Moses Extensions -> Log Varnish Cache Purging Tags

### Output

Once the GraphQLLogging is enabled, The graphql request and response will be available in the below log file `MAGE_ROOT/var/log/moses-logging.log`

Please note that, This extension is for debugging purpose only, Once enabled this extension will write all/partial request and response to the log file based on your configuration, So plz be advised that keeping this extension on enabled mode for a long time will fill up your disk space. Once the debugging is completed this needs to be disabled