



Monitor Laravel with Prometheus + Grafana

Application Monitoring



Grafana



Create a file named indexController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Http\Response;
use Prometheus\CollectorRegistry;
use Prometheus\Storage\InMemory;
use Illuminate\Http\JsonResponse;

use App\Helpers\PrometheusHelper;
use App\Helpers\ResponseService;

class IndexController extends Controller
{
    public function createRequest(Request $request): JsonResponse
    {
        $counter = PrometheusHelper::getOrCreateRegisterCounter('lumen_app', 'lumen_create_req_total', 'Total hit create');
        $counter->inc();

        $data = ['message' => 'Data created successfully'];

        return ResponseService::produceCode(200, 'success', date('Y-m-d H:i:s'), $data);
    }

    public function read($id)
    {
        $counter = PrometheusHelper::getOrCreateRegisterCounter('lumen_app', 'lumen_read_req_total', 'Total hit read');
        $counter->inc();

        $data = ['id' => $id, 'message' => 'Data retrieved successfully'];

        return ResponseService::produceCode(200, 'success', date('Y-m-d H:i:s'), $data);
    }

    public function updateRequest(Request $request, $id)
    {
        $counter = PrometheusHelper::getOrCreateRegisterCounter('lumen_app', 'lumen_update_req_total', 'Total hit update');
        $counter->inc();

        $data = ['id' => $id, 'message' => 'Data updated successfully'];

        return ResponseService::produceCode(200, 'success', date('Y-m-d H:i:s'), $data);
    }

    public function delete($id)
    {
        $counter = PrometheusHelper::getOrCreateRegisterCounter('lumen_app', 'lumen_delete_req_total', 'Total hit delete');
        $counter->inc();

        $data = ['id' => $id, 'message' => 'Data deleted successfully'];

        return ResponseService::produceCode(200, 'success', date('Y-m-d H:i:s'), $data);
    }
}
```

Create a file named MetricsController. php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Http\Response;
use Prometheus\CollectorRegistry;
use Prometheus\Storage\InMemory;
use Prometheus\RenderTextFormat;

use App\Helpers\PrometheusHelper;

class MetricsController extends Controller
{
    public function __construct()
    {
    }

    public function index()
    {
        $renderer = new RenderTextFormat();
        $metrics = PrometheusHelper::getMetrics();

        if (empty($metrics)) {
            return response("No metrics available", 200);
        }

        $result = $renderer->render($metrics);
        return response($result, 200)->header('Content-Type', RenderTextFormat::MIME_TYPE);
    }
}
```

Create a file named PrometheusHelper. php

```
<?php

namespace App\Helpers;

use Prometheus\CollectorRegistry;
use Prometheus\Storage\InMemory;
use Prometheus\Storage\Redis;
use Prometheus\Storage\APC;

class PrometheusHelper
{
    private static $registry = null;

    // Inisialisasi registry jika belum ada
    private static function getRegistry()
    {
        if (self::$registry == null) {
            self::$registry = new CollectorRegistry(new InMemory());
        }
        return self::$registry;
    }

    // Helper untuk mendaftarkan atau mengambil counter
    public static function getOrRegisterCounter($namespace, $name, $help, $labels = [])
    {
        $registry = self::getRegistry();
        return $registry->getOrRegisterCounter($namespace, $name, $help, $labels);
    }

    // Mendapatkan metrik untuk rendering di /metrics endpoint
    public static function getMetrics()
    {
        $metrics = self::getRegistry()->getMetricFamilySamples();
        return $metrics;
    }
}
```

Create a file named ResponseService.php

```
<?php

namespace App\Helpers;

use Carbon\Carbon;
use Illuminate\Http\JsonResponse;


use Auth;

class ResponseService
{
    public static function produceCode($responseCode = 200
                                     , $responseDescription = "success"
                                     , $responseTime = null
                                     , $responseData = null)
    {
        $jsonResponse = new JsonResponse([
            'responseCode' => $responseCode,
            'responseDescription' => $responseDescription,
            'responseTime' => $responseTime,
            'responseData' => $responseData,
        ], $responseCode);


        $jsonResponse->withHeaders([
            'Content-Type' => 'application/json',
        ]);

        return $jsonResponse;
    }
}
```

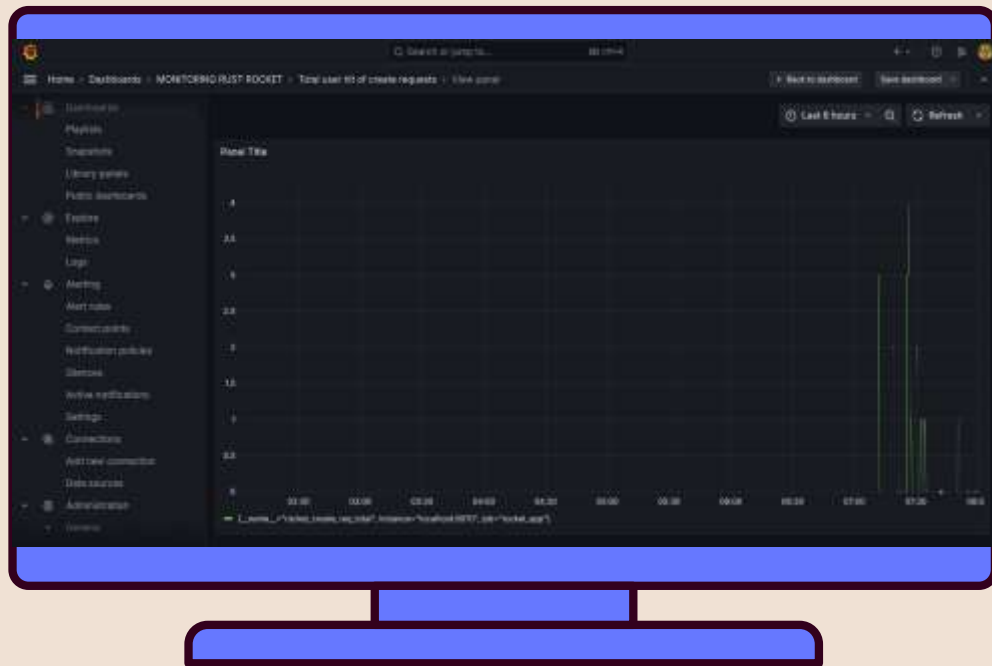
Append this code
to the web.php



```
$router->get('/', function () use ($router) {  
    return $router->app->version();  
});  
  
Route::get('/metrics', 'MetricsController@index');  
  
Route::group(['prefix' => 'api'], function () use ($router)  
{  
    Route::post('/crud', 'IndexController@create');  
    Route::get('/crud/{id}', 'IndexController@read');  
    Route::put('/crud/{id}', 'IndexController@update');  
    Route::delete('/crud/{id}', 'IndexController@delete');  
});
```



GRAFANA MOCKUP



SEE YOU

.....

