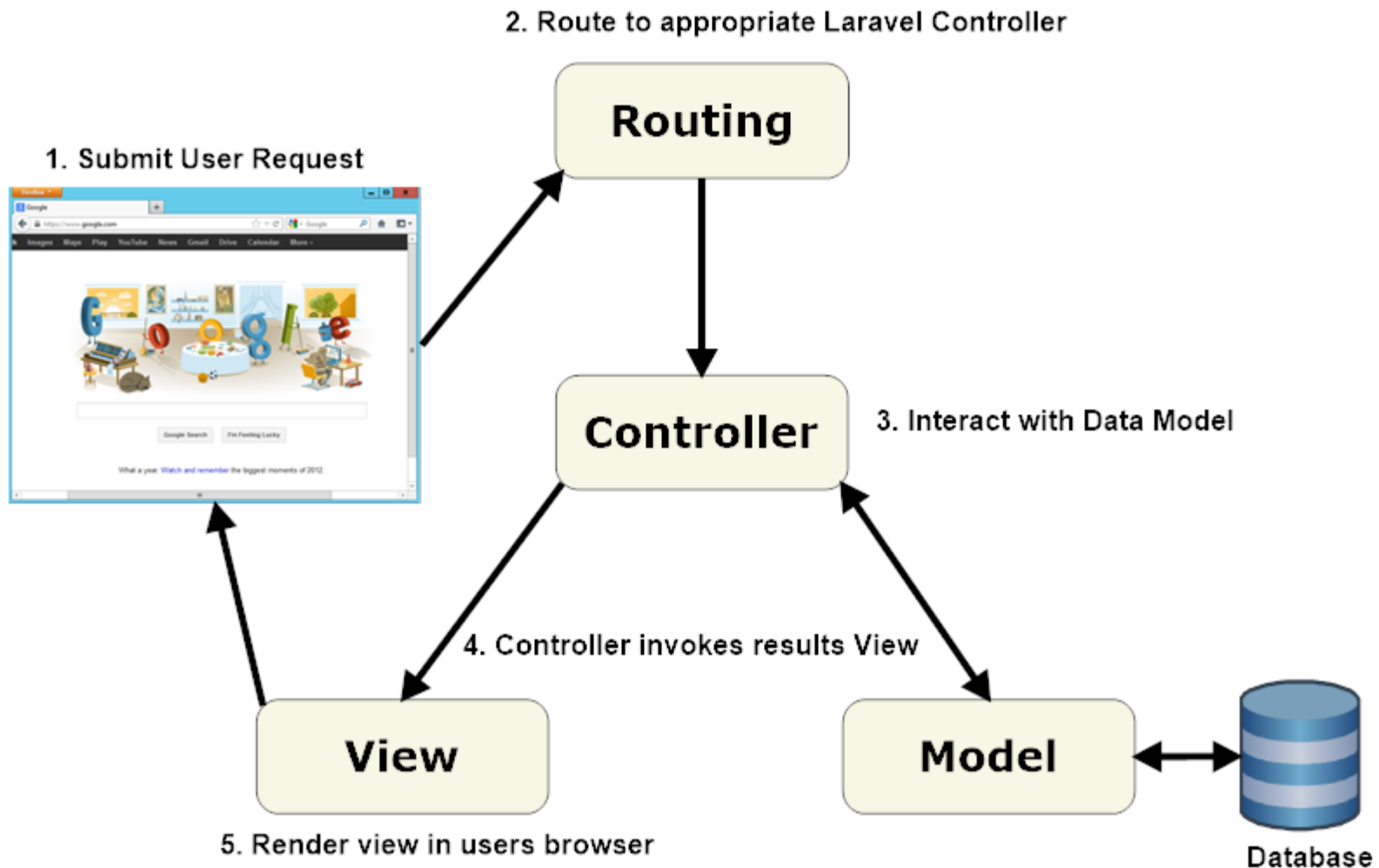


Lecture 6.2
Laravel – Routing
Laravel – Middleware

Course Instructor
Engr. Madeha Mushtaq

Components of Laravel



Basic Routing in Laravel

- Routing is one of the coolest feature of Laravel
- You will define most of the routes for your application in web.php file at `\routes\web.php`
- Web.php is included in every new project.
- Web.php is used to create custom URLs.
- To define a route call the static name `Route::` followed by either a post or get (any can also be used to match both post and get requests) to match the HTTP action.

Basic Routing in Laravel

- Based on the path you can
 - Load a closure function or
 - A controller!
- The most basic Laravel routes simply accept a URL and a Closure function.
- Can be used for static pages and no need to add any controller.

Basic Routing in Laravel

- `Route::get('/', function()
{
 return 'Hello World';
});` OR
- `Route::get('/', function()
{
 return view
 ('welcome');
});`

Closure Function

- Closures are PHP's version of **anonymous functions**.
- They are useful when **you only need simple logic for a route**.
- To use a closure first call `Route::` then set the url pattern you want to match against followed by a function
- `Route::get('simple', function()`
- `{ //do something simple });`

Closure Function

- A simple closure function

```
Route::get('/', function () {  
    return view('welcome');  
});
```

Using a Controller

- Create a controller under app/Http/Controllers.
- Hellocontroller.php

```
HelloController.php
<?php

namespace App\Http\Controllers;
use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class HelloController extends Controller
{
    public function index() {
        return view ('hello');
    }
}
```


Using a Controller

- Define the route in web.php, “Route::get('hello', 'Hellocontroller@index');”

```
<?php

use Illuminate\Support\Facades\Route;

/*
|-----
|  Web Routes
|-----
|
|  Here is where you can register web routes for your application. These
|  routes are loaded by the RouteServiceProvider within a group which
|  contains the "web" middleware group. Now create something great!
|
*/

Route::get('hello', 'Hellocontroller@index');
```

Secured Routing

- `Route::get('myaccount',array('https', function(){
return 'My https secured page'; }));`
- Url: `https://127.0.0.1:8000//myaccount`
 - O/P: My https secured page
- Url: `http://127.0.0.1:8000//myaccount`
 - O/P: Not found exception

Route Parameters

- There are two types of route parameters:
 - Required Parameters
 - Optional Parameters

Required Parameters

- `Route::get('profile/{name}', function($name) {
 return 'Hey there '. $name.' !Welcome to
 whatsapp!';
});`
- Url: `http://127.0.0.1:8000/profile/john`
– O/P : Hey there john! Welcome to whatsapp!
- **Note:** Route parameters cannot contain the - character. Use an underscore (_) instead.

Example

- Web.php

```
<?php

use Illuminate\Support\Facades\Route;

/*
|-----
| Web Routes
|-----
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/

Route::get('hello/{fname}', 'Hellocontroller@index');
```

Example

- Hellocontroller.php

```
<?php
namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class Hellocontroller extends Controller{
    public function index($fname){
        echo "Hello".$fname;
    }
}
?>
```

Example



 127.0.0.1:8000/hello/sara

Hello sara

Example

- An alternative could be without using a controller. We can define the function in web.php

```
Route::get('hello/{fname}', function($fname) {  
    echo $fname;  
});  
  
Route::get('/', function () {  
    return view('welcome');  
});
```


Optional Parameters

- `Route::get('profile/{name?}', function($name = null) {
 return 'Hey there'. $name.'!Welcome to whatsapp!';
});`
- Url: `http://localhost/profile/`
 - O/P : Hey there !Welcome to whatsapp!
- **Optional Route Parameters With Default Value**
- `Route::get('user/{name?}', function($name = 'John') {
 return $name; });`

Example

- Include in web.php

```
Route::get('pricefilter/{max}/{min?}', function($max, $min = '0') {  
    echo "max = ".$max." min = ".$min;  
});
```

Example

← → ↻ ⓘ 127.0.0.1:8000/pricefilter/30/20

max = 30 min = 20

← → ↻ ⓘ 127.0.0.1:8000/pricefilter/30

max = 30 min = 0

And if we don't enter minimum value, we won't get an error.

Route Constraints

- `Route::get('profile/{name?}', function($name = null) {
 return 'Hey there '. $name.' ! Welcome to whatsapp
 !';
})`
- `->where('name', '[A-Za-z]+');`
- Url: `http:// 127.0.0.1:8000/profile/p45`
 - O/P: Redirected to missing page

Route Constraints

- `Route::get('user/{id}', function($id)`
- `{ // })`
- `->where('id', '[0-9]+');`
- **Array Of Constraints:**
- `Route::get('user/{id}/{name}', function($id, $name)`
- `{ // })`
- `->where(['id' => '[0-9]+', 'name' => '[a-z]+']);`

Route Constraints

- Including validation for price:

```
Route::get('demo/{price}', function($price) {  
    echo $price;  
})-> where(['price' => "[0-9]+"]);  
  
Route::get('/', function () {  
    return view('welcome');  
});
```

← → ↻ ⓘ 127.0.0.1:8000/demo/30

Global Pattern

- The same constraint is applied to route parameters all across the web.php
- `Route::pattern('id', '[0-9]+');`
- `Route::get('profile/{id}', function($id){ // Only called if {id} is numeric. });`

Middleware

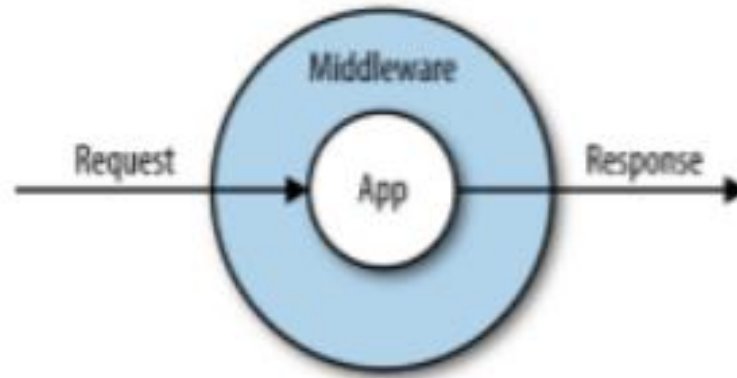
- Powerful piece of software that resides in the middle of our web application architecture.
- Provides high level abstraction
- The word Middleware was first used in the field of distributed systems.

Middleware

- The “software glue” binding together services that allow the client and server to interact.
- The “dash” in client-server.
- It provides services to applications beyond those available from underlying operating systems.
- It allows multiple processes running on different machines to interact, where natively they would not be able to.

Middleware

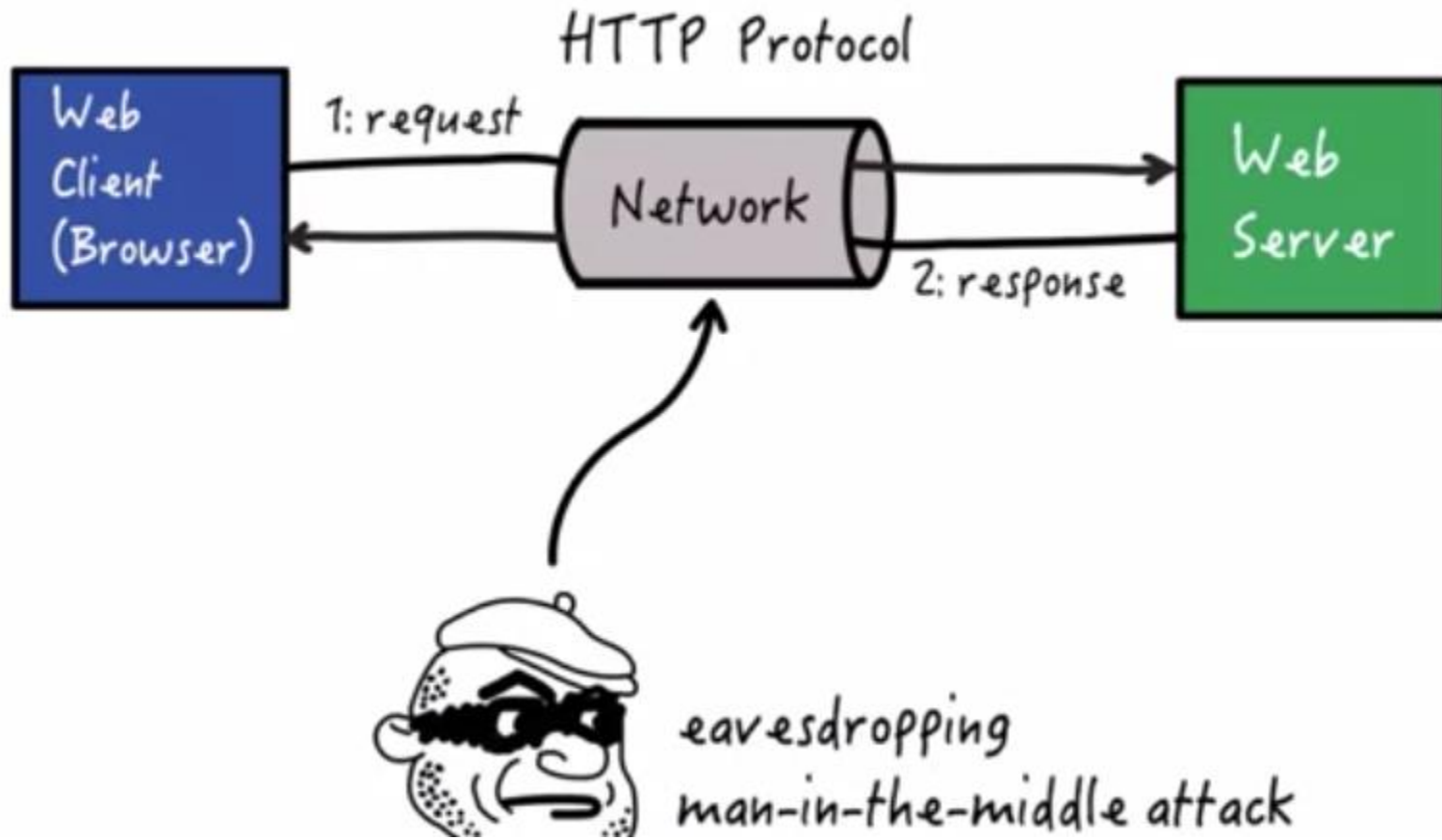
- Middleware provide a convenient mechanism for inspecting and filtering HTTP requests entering your application.



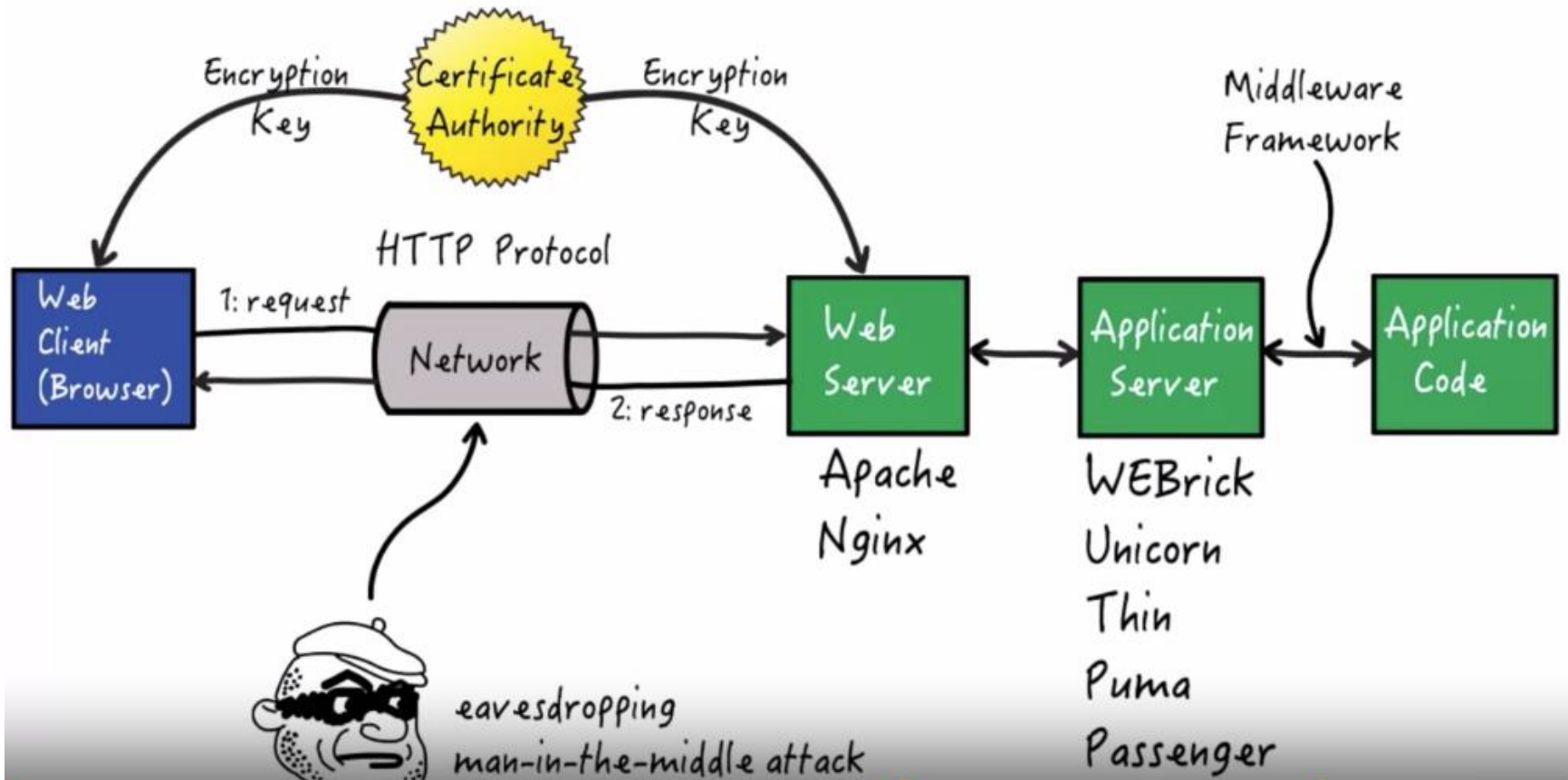
Middleware in Web Apps

- The use of middleware makes it easy for software developers to build web applications.
- Typical middleware components in web applications.
 - Web Servers
 - Applications Servers
 - Middleware Frameworks
 - Session Management Services
 - Proxy Services
 - Load Balancing Services

Middleware in Web Apps



Middleware Frameworks



Middleware Frameworks

- Middleware Frameworks have been created for connecting web application frameworks to application servers.
- Application developers generally don't modify the services provided by the middleware framework.
- Middleware Framework APIs are mainly used by the developers of web app frameworks and are usually not exposed to the actual users of the web app framework.

Laravel Middleware

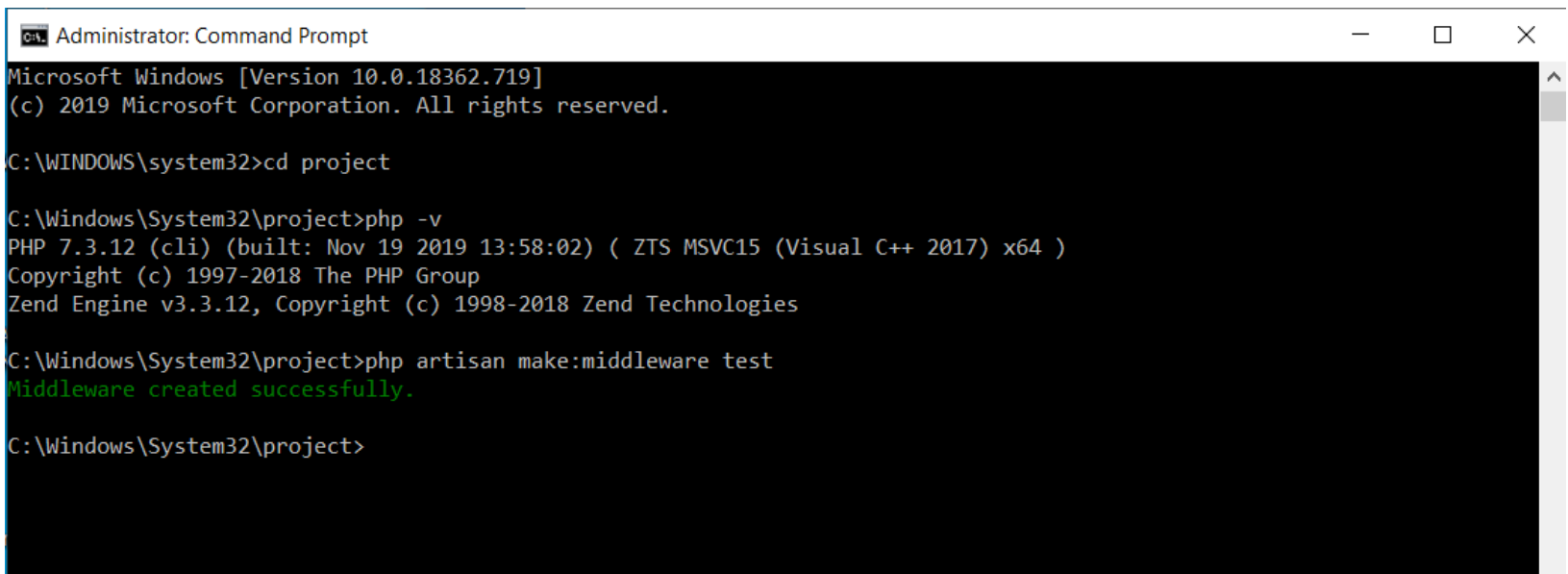
- Laravel includes a middleware that verifies the user of your application is authenticated.
- If the user is not authenticated, the middleware will redirect the user to your application's login screen.
- However, if the user is authenticated, the middleware will allow the request to proceed further into the application.
- Laravel also has a middleware for CSRF protection.
- Additional middleware can be written to perform a variety of tasks.

Laravel Middleware

- If we want to authenticate our HTTP request, we will use middleware.
- Suppose, someone requests the home page of your company's website. With the help of middleware, you can check the ip address, if its not company's ip address, you can redirect it to hello page, if its company's ip address we can load the home page.

Laravel Middleware

- Create middleware using command:
“php artisan make:middleware name”.
- Open middleware saved at `\app\Http\Middleware`.



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.18362.719]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd project

C:\Windows\System32\project>php -v
PHP 7.3.12 (cli) (built: Nov 19 2019 13:58:02) ( ZTS MSVC15 (Visual C++ 2017) x64 )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.3.12, Copyright (c) 1998-2018 Zend Technologies

C:\Windows\System32\project>php artisan make:middleware test
Middleware created successfully.

C:\Windows\System32\project>
```

Laravel Middleware

- In web.php

```
Route::get('hello', function(){  
    return view('hello');  
})->middleware('test');
```

- Then open your middleware and edit it.

```
public function handle($request, Closure $next)  
{  
    $ip = $request -> ip();  
    if($ip == '127.0.0.1')  
    {  
        return redirect('/');  
    }  
  
    return $next($request);  
}
```

If ip is that of our local computer, redirect to the home page, other wise open the hello view.

Laravel Middleware

- Finally register your newly created middleware in kernel.php

```
*/  
protected $routeMiddleware = [  
    'auth' => \App\Http\Middleware\Authenticate::class,  
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,  
    'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,  
    'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,  
    'can' => \Illuminate\Auth\Middleware\Authorize::class,  
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,  
    'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,  
    'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,  
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,  
    'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,  
    'test' => \App\Http\Middleware\test::class,  
];  
}
```

Laravel Middleware

- Run it, you will be redirected to the home page.
- Hard code any IP Address other than the local IP address and you will be redirected to the Hello page.

← → ↻ 127.0.0.1:8000/hello

Laravel