

Name: Esraa Ramadan Abdelghani wahba

What are some common uses of middleware in Laravel

Middleware in Laravel are used to filter HTTP requests entering your application. Some common uses of middleware in Laravel are:

- **Authentication:** Middleware can be used to check if a user is authenticated. For example, if a user tries to access a protected route, the application can check if the user is logged in before proceeding.
- **Redirection:** Middleware can be used to redirect users based on certain conditions. For example, you can create a middleware that checks if a user has a certain role, and if they don't, redirect them to a different page.
- **Logging:** Middleware can be used to log certain events, such as a user logging in or accessing a specific page.
- **CORS (Cross-Origin Resource Sharing):** Middleware can be used to handle CORS requests. For example, if your application serves APIs that need to be accessed by multiple domains, middleware can be used to handle the CORS headers.
- **Language Preference:** Middleware can be used to determine the language preference of a user. For example, if a user's browser is set to a different language, middleware can be used to detect this and change the language of the application accordingly.
- **IP Blocking:** Middleware can be used to block requests from certain IP addresses. For example, if your application needs to restrict access to specific IP addresses, middleware can be used to check the user's IP address and either allow or block access.

Remember that middleware can be global, meaning it applies to all requests, or they can be assigned to specific routes or groups of routes. This gives you a lot of flexibility when deciding where and when to apply your middleware.

Middleware can be created by executing the following command

[php artisan make:middleware <middleware-name>](#)

Example

Registering Middleware

We need to register each and every middleware before using it. There are two types of Middleware in Laravel.

- Global Middleware
- Route Middleware

The **Global Middleware** will run on every HTTP request of the application, whereas the **Route Middleware** will be assigned to a specific route. The middleware can be registered at **app/Http/Kernel.php**. This file contains two properties **\$middleware** and **\$routeMiddleware**. **\$middleware** property is used to register Global Middleware and **\$routeMiddleware** property is used to register route specific middleware.

To register the global middleware, list the class at the end of **\$middleware** property.

```
protected $middleware = [

    \Illuminate\Foundation\Http\Middleware\CheckForMaintenanceMode::class,
        \App\Http\Middleware\EncryptCookies::class,

    \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,

    \Illuminate\Session\Middleware\StartSession::class,

    \Illuminate\View\Middleware\ShareErrorsFromSession::class,
        \App\Http\Middleware\VerifyCsrfToken::class,
];
```

To register the route specific middleware, add the key and value to **\$routeMiddleware** property.

```
protected $routeMiddleware = [
    'auth' =>
        \App\Http\Middleware\Authenticate::class,
    'auth.basic' =>
        \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'guest' =>
        \App\Http\Middleware\RedirectIfAuthenticated::class
];
```

Example

We have created **AgeMiddleware** in the previous example. We can now register it in route specific middleware property. The code for that registration is shown below.

The following is the code for **app/Http/Kernel.php** –

```
<?php

namespace App\Http;
use Illuminate\Foundation\Http\Kernel as
HttpKernel;

class Kernel extends HttpKernel {
    protected $middleware = [

        \Illuminate\Foundation\Http\Middleware\CheckForMain
tenanceMode::class,
        \App\Http\Middleware\EncryptCookies::class,

        \Illuminate\Cookie\Middleware\AddQueuedCookiesToRes
ponse::class,

        \Illuminate\Session\Middleware\StartSession::class,

        \Illuminate\View\Middleware\ShareErrorsFromSession:
:class,
        \App\Http\Middleware\VerifyCsrfToken::class,
    ];

    protected $routeMiddleware = [
        'auth' =>
        \App\Http\Middleware\Authenticate::class,
        'auth.basic' =>
        \Illuminate\Auth\Middleware\AuthenticateWithBasicAu
th::class,
        'guest' =>
        \App\Http\Middleware\RedirectIfAuthenticated::class
    ,
        'Age' =>
        \App\Http\Middleware\AgeMiddleware::class,
    ];
}
```

Middleware Parameters

We can also pass parameters with the Middleware. For example, if your application has different roles like user, admin, super admin etc. and you want to authenticate the action based on role, this can be achieved by passing parameters with middleware. The middleware that we create contains the following function and we can pass our custom argument after the **\$next** argument.

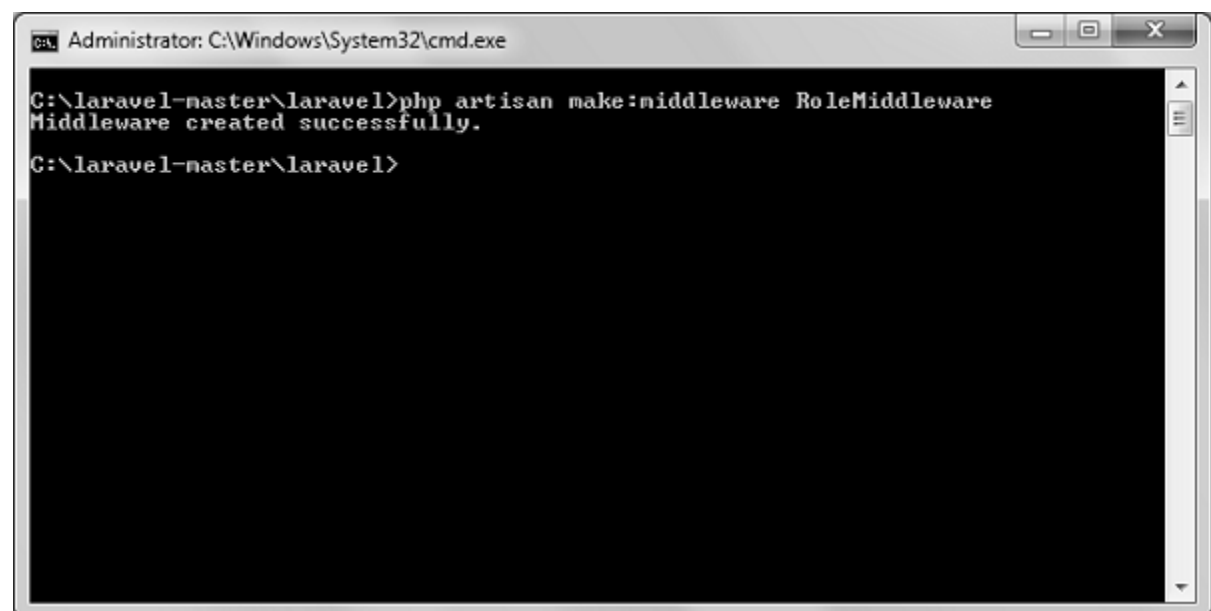
```
public function handle($request, Closure $next) {  
    return $next($request);  
}
```

Example

Step 1 – Create RoleMiddleware by executing the following command –

```
php artisan make:middleware RoleMiddleware
```

Step 2 – After successful execution, you will receive the following output –

A screenshot of a Windows command prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The window shows the following text:

```
C:\laravel-master\laravel>php artisan make:middleware RoleMiddleware  
Middleware created successfully.  
C:\laravel-master\laravel>
```

Step 3 – Add the following code in the handle method of the newly created RoleMiddlewareat **app/Http/Middleware/RoleMiddleware.php**.

```
<?php
```

```

namespace App\Http\Middleware;
use Closure;

class RoleMiddleware {
    public function handle($request, Closure $next,
$role) {
        echo "Role: ".$role;
        return $next($request);
    }
}

```

Step 4 – Register the RoleMiddleware
in **app\Http\Kernel.php** file. Add the line highlighted in gray color in that file to register RoleMiddleware.

```

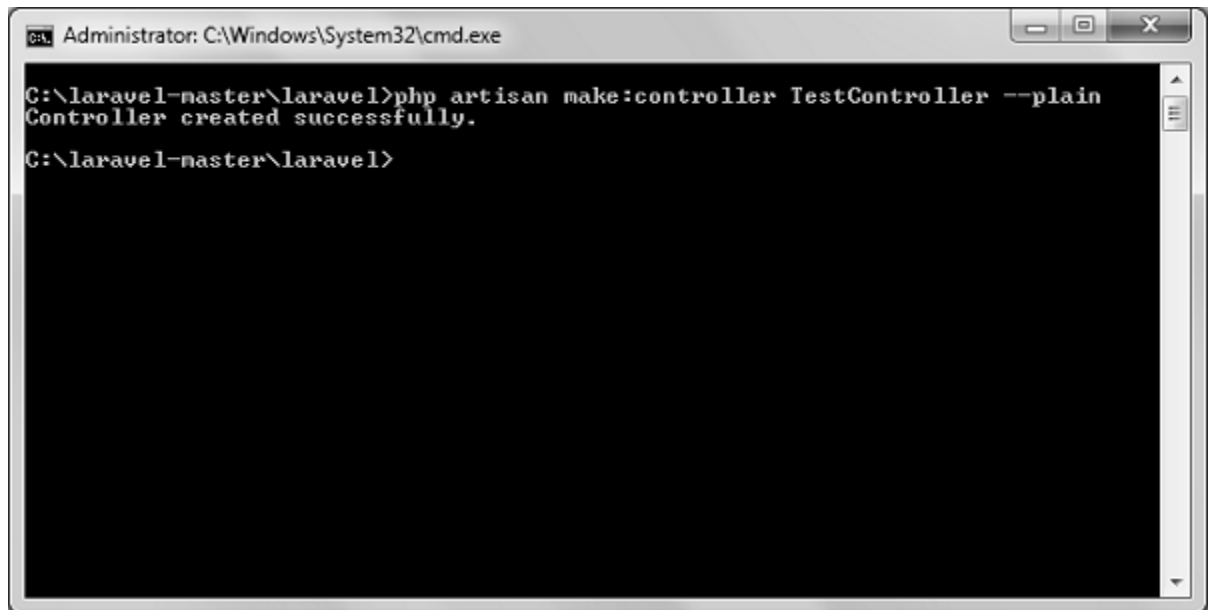
/**
 * The application's route middleware.
 *
 * @var array
 */
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
    'Age' => \App\Http\Middleware\AgeMiddleware::class,
    'After' => \App\Http\Middleware\AfterMiddleware::class,
    'Before' => \App\Http\Middleware\BeforeMiddleware::class,
    'First' => \App\Http\Middleware\FirstMiddleware::class,
    'Second' => \App\Http\Middleware\SecondMiddleware::class,
    'Role' => \App\Http\Middleware\RoleMiddleware::class,
];

```

Step 5 – Execute the following command to create TestController –

```
php artisan make:controller TestController --plain
```

Step 6 – After successful execution of the above step, you will receive the following output –

A screenshot of a Windows command prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The window shows the following text:

```
C:\laravel-master\laravel>php artisan make:controller TestController --plain
Controller created successfully.
C:\laravel-master\laravel>
```

Step 7 – Copy the following lines of code to `app/Http/TestController.php` file.

`app/Http/TestController.php`

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Http\Requests;
use App\Http\Controllers\Controller;

class TestController extends Controller {
    public function index() {
        echo "<br>Test Controller.";
    }
}
```

Step 8 – Add the following line of code in `app/Http/routes.php` file.

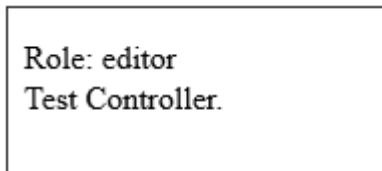
`app/Http/routes.php`

```
Route::get('role',[
    'middleware' => 'Role:editor',
    'uses' => 'TestController@index',
]);
```

Step 9 – Visit the following URL to test the Middleware with parameters

<http://localhost:8000/role>

Step 10 – The output will appear as shown in the following image.



Terminable Middleware

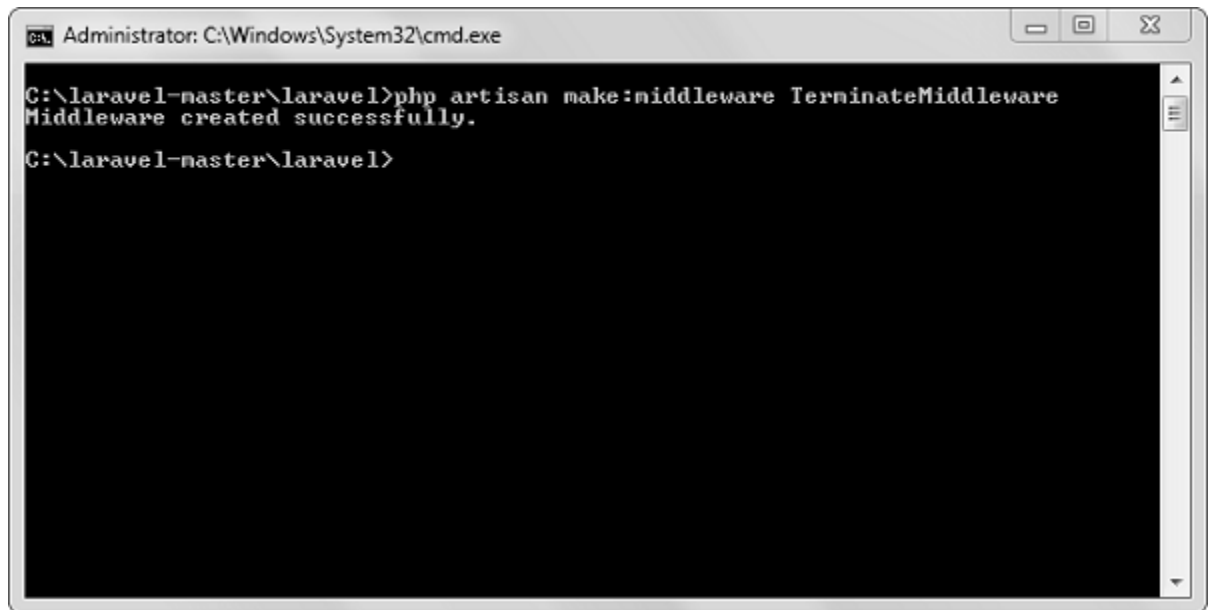
Terminable middleware performs some task after the response has been sent to the browser. This can be accomplished by creating a middleware with **terminate** method in the middleware. Terminable middleware should be registered with global middleware. The terminate method will receive two arguments **\$request** and **\$response**. Terminate method can be created as shown in the following code.

Example

Step 1 – Create **TerminateMiddleware** by executing the below command.

```
php artisan make:middleware TerminateMiddleware
```

Step 2 – The above step will produce the following output –

A screenshot of a Windows command prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The window shows the following text:

```
C:\laravel-master\laravel>php artisan make:middleware TerminateMiddleware
Middleware created successfully.
C:\laravel-master\laravel>
```

Step 3 – Copy the following code in the newly created `TerminateMiddleware` at `app/Http/Middleware/TerminateMiddleware.php`.

```
<?php

namespace App\Http\Middleware;
use Closure;

class TerminateMiddleware {
    public function handle($request, Closure $next)
    {
        echo "Executing statements of handle method
of TerminateMiddleware.";
        return $next($request);
    }

    public function terminate($request, $response) {
        echo "<br>Executing statements of terminate
method of TerminateMiddleware.";
    }
}
```

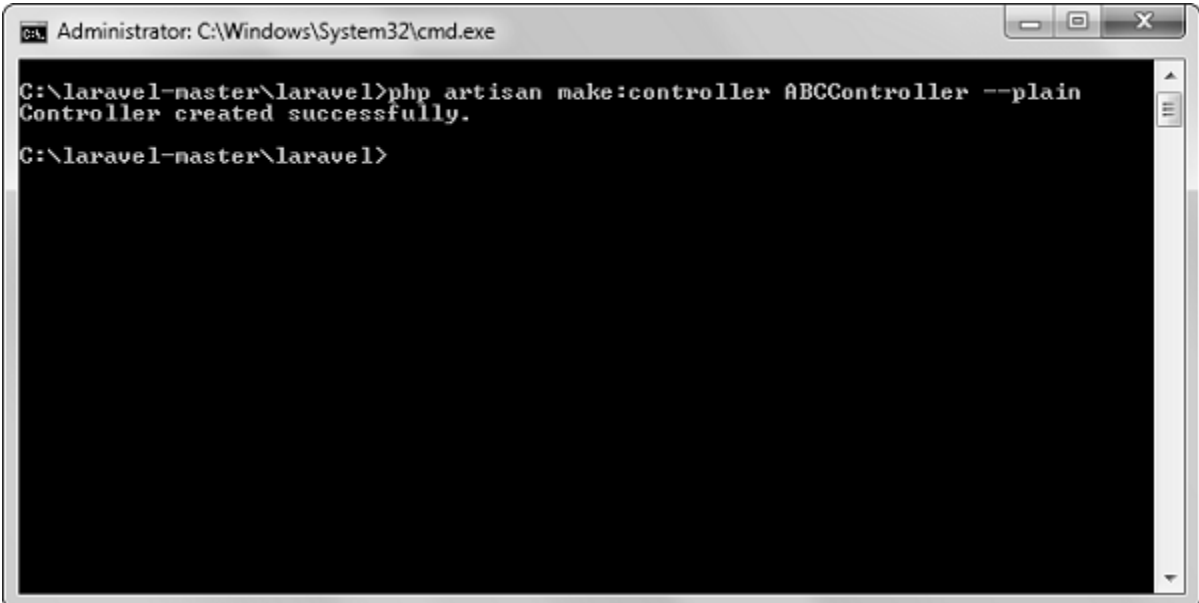
Step 4 – Register
the `TerminateMiddleware` in `app/Http/Kernel.php` file. Add the line highlighted in gray color in that file to register `TerminateMiddleware`.


```
/**
 * The application's route middleware.
 *
 * @var array
 */
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
    'Age' => \App\Http\Middleware\AgeMiddleware::class,
    'After' => \App\Http\Middleware\AfterMiddleware::class,
    'Before' => \App\Http\Middleware\BeforeMiddleware::class,
    'First' => \App\Http\Middleware\FirstMiddleware::class,
    'Second' => \App\Http\Middleware\SecondMiddleware::class,
    'Role' => \App\Http\Middleware\RoleMiddleware::class,
    'terminate' => \App\Http\Middleware\TerminateMiddleware::class,
];
```

Step 5 – Execute the following command to create **ABCController**.

```
php artisan make:controller ABCController --plain
```

Step 6 – After the successful execution of the URL, you will receive the following output –



```
Administrator: C:\Windows\System32\cmd.exe

C:\laravel-master\laravel>php artisan make:controller ABCController --plain
Controller created successfully.

C:\laravel-master\laravel>
```

Step 7 – Copy the following code to **app/Http/ABCController.php** file.

app/Http/ABCController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Http\Requests;
use App\Http\Controllers\Controller;

class ABCController extends Controller {
    public function index() {
        echo "<br>ABC Controller.";
    }
}
```

Step 8 – Add the following line of code in **app/Http/routes.php** file.

app/Http/routes.php

```
Route::get('terminate',[
    'middleware' => 'terminate',
    'uses' => 'ABCController@index',
]);
```

Step 9 – Visit the following URL to test the Terminable Middleware.

<http://localhost:8000/terminate>

Step 10 – The output will appear as shown in the following image

```
Install Auth Scaffold
composer require laravel/ui
C:\xampp\htdocs\laravel\my-app>composer require laravel/ui

generate auth scaffold with bootstrap
php artisan ui bootstrap --auth
C:\xampp\htdocs\laravel\my-app>php artisan ui bootstrap --auth

install npm packages
npm install
C:\xampp\htdocs\laravel\my-app>npm install

built bootstrap CSS
```

npm run build
C:\xampp\htdocs\laravel\my-app>npm run build

Setting up migration and model

Open the creates_users_table.php migration file Database/migration and update the following field

```
1
2
3 //database/migrations/create_users_table.php
4 <?php
5 use Illuminate\Database\Migrations\Migration;
6 use Illuminate\Database\Schema\Blueprint;
7 use Illuminate\Support\Facades\Schema;
8
9 return new class extends Migration
10 {
11     public function up()
12     {
13         Schema::create('users', function (Blueprint $table) {
14             $table->id();
15             $table->string('name');
16             $table->string('email')->unique();
17             $table->timestamp('email_verified_at')->nullable();
18             $table->string('password');
19             $table->boolean('type')->default(false); //add type boolean Users: 0
20             $table->rememberToken();
21             $table->timestamps();
22         });
23     }
24
25     public function down()
26     {
27         Schema::dropIfExists('users');
28     }
29 };
30
```

run this migration

C:\xampp\htdocs\laravel\my-app>php artisan migrate
open app/User.php and update the below field

app/Models/User.php

```
1
2 //app/Models/User.php
3 <?php
4 namespace App\Models;
5
6 // use Illuminate\Contracts\Auth\MustVerifyEmail;
7 use Illuminate\Database\Eloquent\Factories\HasFactory;
8 use Illuminate\Foundation\Auth\User as Authenticatable;
9 use Illuminate\Notifications\Notifiable;
10 use Laravel\Sanctum\HasApiTokens;
11
12 use Illuminate\Database\Eloquent\Casts\Attribute;
```

```

12
13 class User extends Authenticatable
14 {
15     use HasApiTokens, HasFactory, Notifiable;
16
17     /**
18      * The attributes that are mass assignable.
19      *
20      * @var array<int, string>
21      */
22     protected $fillable = [
23         'name',
24         'email',
25         'password',
26         'type'
27     ];
28
29     protected $hidden = [
30         'password',
31         'remember_token',
32     ];
33
34     protected $casts = [
35         'email_verified_at' => 'datetime',
36     ];
37
38     /**
39      * Interact with the user's first name.
40      *
41      * @param string $value
42      * @return \Illuminate\Database\Eloquent\Casts\Attribute
43      */
44     protected function type(): Attribute
45     {
46         return new Attribute(
47             get: fn ($value) => ["user", "admin", "manager"][$value],
48         );
49     }
50 }
51

```

Create Middleware for checking the users who can access the admin panel or who can access the normal user panel.

php artisan make:middleware UserAccess

C:\xampp\htdocs\laravel\my-app>php artisan make:middleware UserAccess

open app/Http/middleware/UserAccess.php

```

1 //app/Http/middleware/UserAccess.php
2 <?php
3
4 namespace App\Http\Middleware;
5
6 use Closure;

```

```

6 use Illuminate\Http\Request;
7
8 class UserAccess
9 {
10     /**
11      * Handle an incoming request.
12      *
13      * @param \Illuminate\Http\Request $request
14      * @param \Closure(\Illuminate\Http\Request):
15      * (\Illuminate\Http\Response|\Illuminate\Http\RedirectResponse) $next
16      * @return \Illuminate\Http\Response|\Illuminate\Http\RedirectResponse
17      */
18     public function handle(Request $request, Closure $next, $userType)
19     {
20         if(auth()->user()->type == $userType){
21             return $next($request);
22         }
23         return response()->json(['You do not have permission to access for this
24         /* return response()->view('errors.check-permission'); */
25     }
26 }
27
28

```

register this middleware in the app/Http/Kernel.php
add the following \$routeMiddleware property

```

protected $routeMiddleware = [
    'user-access' => \App\Http\Middleware\UserAccess::class,
];

```

Controller

open the app/Http/Controllers/HomeController.php add code

```

2
1 //app\Http\Controllers\HomeController.php
2 <?php
3
4 namespace App\Http\Controllers;
5
6 use Illuminate\Http\Request;
7
8 class HomeController extends Controller
9 {
10     /**
11      * Create a new controller instance.
12      *
13      * @return void
14      */
15     public function __construct()
16     {
17         $this->middleware('auth');
18     }
19
20     /**
21      * Show the application dashboard.
22      *

```

```

20     * @return \Illuminate\Contracts\Support\Renderable
21     */
22     public function index()
23     {
24         return view('home');
25     }
26
27     /**
28      * Show the application dashboard.
29      *
30      * @return \Illuminate\Contracts\Support\Renderable
31      */
32     public function adminHome()
33     {
34         return view('adminHome');
35     }
36
37     /**
38      * Show the application dashboard.
39      *
40      * @return \Illuminate\Contracts\Support\Renderable
41      */
42     public function managerHome()
43     {
44         return view('managerHome');
45     }
46 }

```

Create Blade View

Open the resources/views/home.blade.php file and update the below code.

```

2
1 //resources/views/home.blade.php
2 @extends('layouts.app')
3
4 @section('content')
5 <div class="container">
6     <div class="row justify-content-center">
7         <div class="col-md-8">
8             <div class="card">
9                 <div class="card-header">{{ __('Dashboard') }}</div>
10
11                 <div class="card-body">
12                     @if(auth()->user()->is_admin == 1)
13                     <a href="{{url('admin/routes')}}">Admin</a>
14                     @else
15                     <div class="panel-heading">Normal User</div>
16                     @endif
17                 </div>
18             </div>
19         </div>
20     </div>
21 </div>

```

19 @endsection

20

21

22

Create adminHome.blade.php file inside resources/views/adminHome.blade.php

[?](#)

1

2 //resources/views/adminHome.blade.php

3 @extends('layouts.app')

4

5 @section('content')

6 <div class="container">

7 <div class="row justify-content-center">

8 <div class="col-md-8">

9 <div class="card">

10 <div class="card-header">{{ __('Dashboard') }}</div>

11

12 <div class="card-body">

13 You are a Admin User.

14 </div>

15 </div>

16 </div>

17 @endsection

18

Create managerHome.blade.php file inside resources/views/managerHome.blade.php

directory

[?](#)

1

2

3

4

5 //resources/views/managerHome.blade.php

6 @extends('layouts.app')

7 @section('content')

8 <div class="container">

9 <div class="row justify-content-center">

10 <div class="col-md-8">

11 <div class="card">

12 <div class="card-header">{{ __('Dashboard') }}</div>

13

14 <div class="card-body">

15 You are a Manager User.

16 </div>

17 </div>

18 </div>

19 @endsection

20

21

22

Update LoginController app/Http/Controllers/Auth/LoginController.php

[?](#)

1

2

3

4

5

//app/Http/Controllers/Auth/LoginController.php

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;

```

6 use App\Providers\RouteServiceProvider;
7 use Illuminate\Foundation\Auth\AuthenticatesUsers;
8
9 use Illuminate\Http\Request;
10
11 class LoginController extends Controller
12 {
13     use AuthenticatesUsers;
14
15     protected $redirectTo = RouteServiceProvider::HOME;
16
17     public function __construct()
18     {
19         $this->middleware('guest')->except('logout');
20     }
21
22     public function login(Request $request)
23     {
24         $input = $request->all();
25
26         $this->validate($request, [
27             'email' => 'required|email',
28             'password' => 'required',
29         ]);
30
31         if (auth()->attempt(array('email' => $input['email'], 'password' =>
32 $input['password'])))
33         {
34             if (auth()->user()->type == 'admin') {
35                 return redirect()->route('admin.home');
36             }else if (auth()->user()->type == 'manager') {
37                 return redirect()->route('manager.home');
38             }else{
39                 return redirect()->route('home');
40             }
41         }else{
42             return redirect()->route('login')
43                 ->with('error','Email-Address And Password Are Wrong.');
```

Define Route routes/web.php

```

2 //routes/web.php
3 <?php
4
5 use Illuminate\Support\Facades\Route;
6
7 use App\Http\Controllers\HomeController;
```



```
6
7 Route::get('/', function () {
8     return view('welcome');
9 });
10 Auth::routes();
11
12 //Normal Users Routes List
13 Route::middleware(['auth', 'user-access:user'])->group(function () {
14
15     Route::get('/home', [HomeController::class, 'index'])->name('home');
16 });
17
18 //Admin Routes List
19 Route::middleware(['auth', 'user-access:admin'])->group(function () {
20
21     Route::get('/admin/home', [HomeController::class, 'adminHome'])->
22     >name('admin.home');
23 });
24
25 //Admin Routes List
26 Route::middleware(['auth', 'user-access:manager'])->group(function () {
27
28     Route::get('/manager/home', [HomeController::class, 'managerHome'])->
29     >name('manager.home');
30 });
31
32 Run C:\xampp\htdocs\laravel\my-app>php artisan serve
```