

DOCUMENTATION

CREATING MIDDLEWARE

```
C:\Users\Miranda-Benedicto\Herd\GroupProject2>php artisan make:middleware CheckAge

INFO Middleware [C:\Users\Miranda-Benedicto\Herd\GroupProject2\app\Http\Middleware\
CheckAge.php] created successfully.

C:\Users\Miranda-Benedicto\Herd\GroupProject2>php artisan make:middleware LogRequest

INFO Middleware [C:\Users\Miranda-Benedicto\Herd\GroupProject2\app\Http\Middleware\
LogRequest.php] created successfully.
```

We make a CheckAge and LogRequest middleware in Laravel project using the command prompt. First, we ran the command php artisan make:middleware CheckAge to create the CheckAge middleware, which verifies the user’s age and manages redirection based on specific age rules. Then, we used php artisan make:middleware LogRequest to produce the LogRequests middleware, which logs information about incoming HTTP requests, such as the request method and full URL.

CheckAge.php

```
6 class CheckAge
7 {
8     /**
9      * Handle an incoming request.
10     *
11     * @param \Illuminate\Http\Request $request
12     * @param \Closure $next
13     * @return mixed
14     */
15     public function handle(Request $request, Closure $next)
16     {
17         \Log::info('CheckAge middleware is invoked');
18         $age = session('age', 0);
19         \Log::info("Age in middleware: $age"); // Log the age value
20
21         // Validate age
22         if ($age < 18) {
23             // Redirect to access denied page for users below 18
24             return redirect()->route('accessDenied');
25         } elseif ($age >= 18 && $age <= 24) {
26             // Allow access to full page but not the restricted page
27             if ($request->is('restricted')) {
28                 return redirect()->route('accessDenied');
29             }
30         } elseif ($age >= 25) {
31             // Allow access to all pages for users 25 and above
32         }
33         return $next($request); // Proceed to the requested page
34     }
35 }
```

This code is a middleware named CheckAge that controls access to different pages of a website based on the user's age. When a request is made to the website, the handle method is triggered, which retrieves the user's age from the session. If the age is not stored, it defaults to 0. The code then checks the user's age and applies specific rules: users under 18 are redirected to an "access denied" page; users aged 18 to 24 can access most pages, but if they try to visit a restricted page, they are also redirected; and users aged 25 and above have full access to all pages. The middleware logs the age and the fact that it has been triggered, making it easier to track in the logs. If the user passes all checks, the middleware allows the request to proceed to the requested page.

LogRequest.php

This code sets up a middleware in Laravel called LogRequest, which tracks details of each request made to the website. When someone makes a request (like visiting a page), the middleware runs and collects information such as the date and time, the type of request (like GET or POST), and the full web address. It then logs this information into a custom log file. After logging, the request is allowed to move forward so the user can see the page or data they asked for.

```
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Closure;
6 use Illuminate\Http\Request;
7 use Illuminate\Support\Facades\Log;
8
9 class LogRequest
10 {
11     /**
12     * Handle an incoming request.
13     *
14     * @param \Illuminate\Http\Request $request
15     * @param \Closure $next
16     * @return mixed
17     */
18     public function handle(Request $request, Closure $next)
19     {
20         $logData = '[' . now() . ']' . ' ' . $request->method() . ' ' . $request->fullUrl();
21         Log::channel('custom')->info($logData);
22
23         return $next($request);
24     }
25 }
```

config/logging.php

To make the LogRequests middleware work and save its output in log.txt, we created a custom log channel in the logging.php file. We set up the 'custom' channel to use a 'single' driver, which means that all log entries from the LogRequests middleware will be saved in log.txt located in the storage/logs folder. This way, all logged details, like URLs and timestamps, are stored in this specific file.

```
web.php  CheckAge.php  logging.php  # style.css
config > logging.php
8 return [
53     'channels' => [
124         ],
125
126         'emergency' => [
127             'path' => storage_path('logs/laravel.log'),
128         ],
129
130         // Add this part for the custom log channel
131         'custom' => [
132             'driver' => 'single',
133             'path' => storage_path('logs/log.txt'),
134             'level' => 'info',
135         ],
136     ],
137 ];
138
```

REGISTRATION IN Kernel.php

```
web.php Kernel.php X CheckAge.php logging.php X # style.css
app > Http > Kernel.php
1 <?php
2
3 namespace App\Http\Kernel;
4
5 use Illuminate\Foundation\Http\Kernel as HttpKernel;
6
7 class Kernel extends HttpKernel
8 {
9     /**
10      * The application's request middleware.
11      *
12      * These middleware are run during every request to your application.
13      *
14      * @var array<int, class-string|string>
15      */
16     protected $middleware = [
17         //global middleware part added
18         \App\Http\Middleware\CheckAge::class,
19         \App\Http\Middleware\LogRequest::class,
20     ];
21
22     /**
23      * The application's route middleware.
24      *
25      * These middleware may be assigned to groups or used individually.
26      *
27      * @var array<string, class-string|string>
28      */
29     protected $routeMiddleware = [
30         'checkAge' => \App\Http\Middleware\CheckAge::class,
31     ];
32 }
```

Since we couldn't locate the Kernel.php file in the App/Http directory, we decided to manually create a new file there to manage our middleware, which is important for handling middleware tasks in the application. We registered the CheckAge.php middleware for specific routes that need age verification. On the other hand, we set up the LogRequests.php middleware as global middleware to make sure that all incoming HTTP requests are logged consistently across the entire application, no matter which route is accessed. In summary, this code sets up the Kernel class to manage both global and route-specific middleware in the application.

ROUTING CONFIGURATION:

```
web.php X Kernel.php CheckAge.php logging.php # style.css
routes > web.php
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4 use Illuminate\Http\Request;
5 use App\Http\Middleware\CheckAge;
6 use App\Http\Middleware\LogRequest;
7
8 // Group routes that use LogRequest middleware
9 Route::middleware([LogRequest::class])->group(function () {
10
11     // Login route
12     Route::get('/', function () {
13         return view('login');
14     })->name('login');
15
16     // Route to handle the username and age submission, and store them in the session
17     Route::post('/set-username', function (Request $request) {
18         $username = $request->input('username') ?: 'Guest';
19         $age = $request->input('age');
20
21         // Store the username and age in the session
22         session(['username' => $username, 'age' => $age]);
23
24         // Redirect to the home page
25         return redirect()->route('home');
26     })->name('set-username');
27
28     // About page route
29     Route::get('/about', function () {
30         $username = session('username', 'Guest');
31         return view('about', ['username' => $username]);
32     })->name('about');
```


```
web.php X Kernel.php CheckAge.php logging.php # style.css
routes > web.php
9 Route::middleware([LogRequest::class])->group(function () {
35     Route::get('/content', function () {
36         $username = session('username', 'Guest');
37         return view('content', ['username' => $username]);
38     })->name('content');
39
40     // Contact Us page route
41     Route::get('/contact', function () {
42         $username = session('username', 'Guest');
43         return view('contact', ['username' => $username]);
44     })->name('contact');
45
46     // Access Denied route
47     Route::get('/accessDenied', function () {
48         \Log::info('Access Denied page visited');
49         return view('accessDenied'); // Ensure you create the 'accessDenied'
50     })->name('accessDenied');
51 });
52
53 // Home route for users 18 and above (excluding 25 and above)
54 Route::get('/home', function () {
55     $username = session('username', 'Guest');
56     $age = session('age', 0); // Retrieve age for logging/debugging
57     \Log::info("Username: $username, Age: $age");
58     return view('home', ['username' => $username]);
59 })->middleware([LogRequest::class, CheckAge::class])->name('home');
60
61 // Restricted page for users 25 and above
62 Route::get('/restricted', function () {
63     $username = session('username', 'Guest');
64     return view('restricted', ['username' => $username]);
65 })->middleware([LogRequest::class, CheckAge::class])->name('restricted');
```

This code configures routing it defines how users can access various pages. It begins by grouping routes that use the LogRequest middleware, which records details about each request. Within this group, several routes are set up: the home page (/) displays the login view; a route (/set-username) handles the submission of the username and age from a form, saves this information in the session, and redirects users to the home page (/home); and routes for the About, Content, and Contact pages retrieve the username from the session and display the corresponding views. Additionally, there's an Access Denied route (/accessDenied) that logs access attempts and shows the accessDenied view. Following these grouped routes, the home route (/home) is protected by both LogRequest and CheckAge middleware, ensuring that only users aged 18 and older can access it. This route retrieves the username and age from the session, logs the information, and displays the home view. Lastly, the restricted page route (/restricted) also employs the same middleware to permit access only to users 25 and older, showing the restricted view along with the username. Overall, this code effectively controls user access based on age and logs important information for all requests.

WEB PAGES

With CheckAge Middleware

This is our login.blade.php file, we just add a form for the age before the users access our website to validate if there age is valid.



Hello!


Please enter your username:

Please enter your age:

Purrfect

Age input (0-17 years old)


This is our accessDenied.blade.php file. If the user enters an age below 18, they will be redirected to this page because they do not meet the age requirement to access our website.



We are Sorry...

The site you're trying to access has restricted access.

Go Back




Hello!

Please enter your username:

Please enter your age:

Purrfect

Age input (18-24 years old)

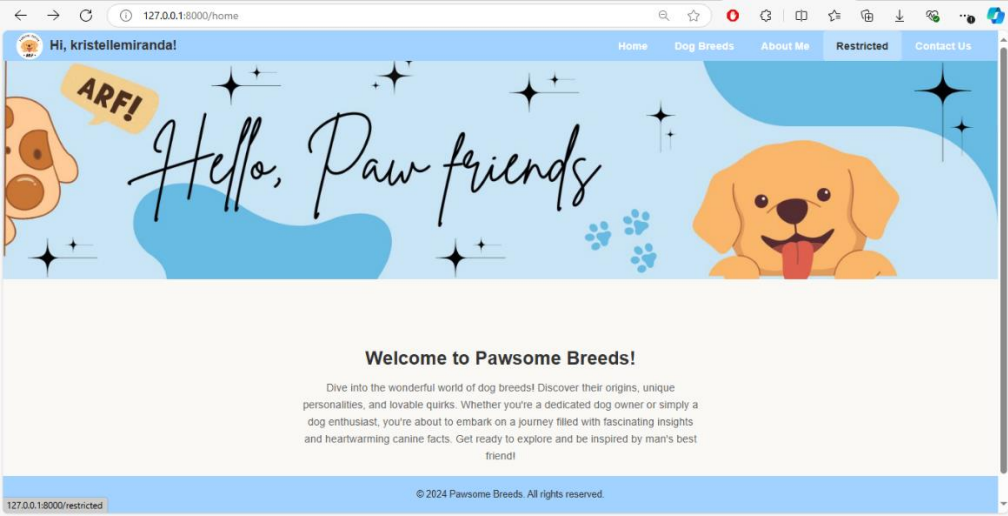


Hello!

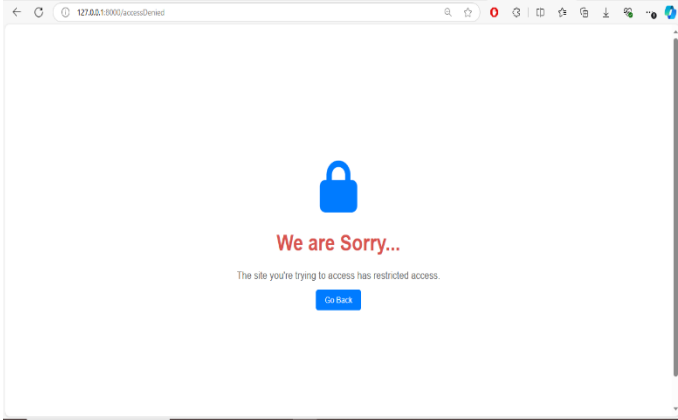
Please enter your username:

Please enter your age:


Purrfect



This is our login input form in the login.blade.php and the welcome page in the home.blade.php. If the user meets the minimum age requirement of 18, they will have an access to the pages but not the restricted page because if they click it it will go to the access denied page because it didn't satisfy the condition to go in restricted page.



Age input (25 above)

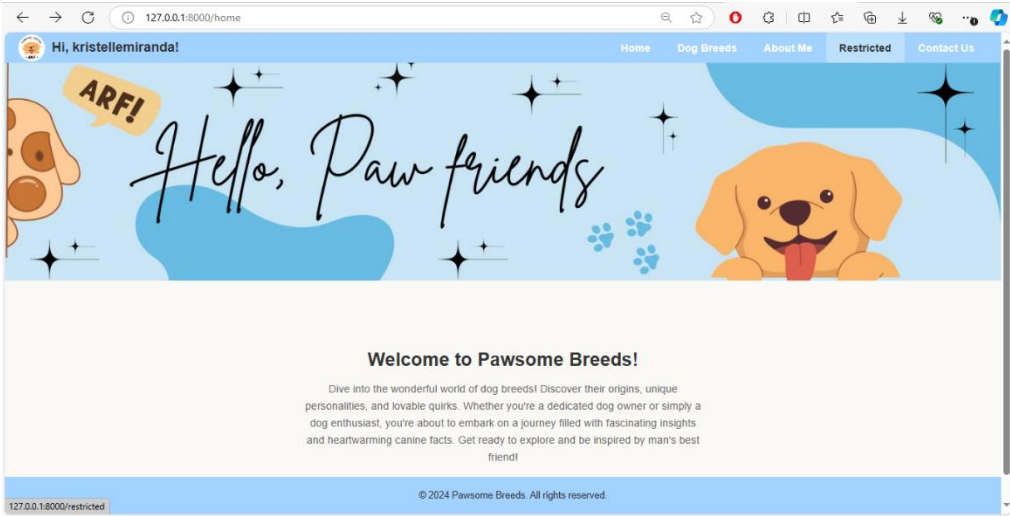


Hello!

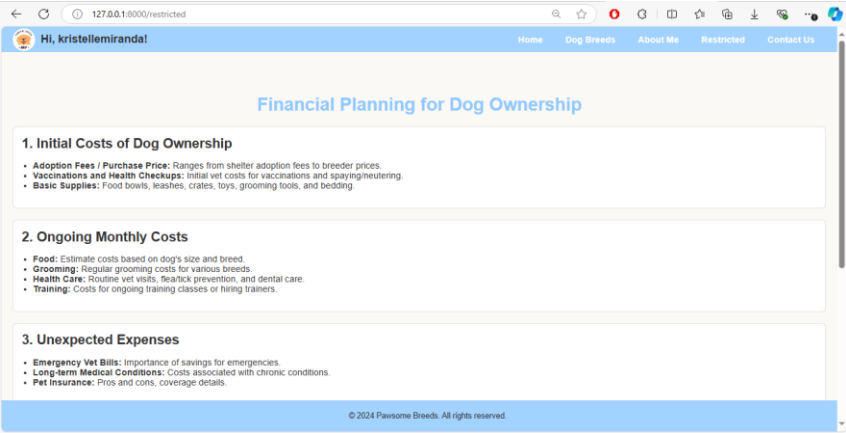
Please enter your username:

Please enter your age:

Purrfect



This is our login input form in the login.blade.php and the welcome page in the home.blade.php. which is a specialized page for users who meet the age requirement of 25 and above.If the user meets the age requirement of 25, they will have an access to all the pages including the restricted page. It's as if users in this age group have more privileges than the rest.



With LogRequests Middleware

web.phplog.txtKernel.phpCheckAge.phplogging.phpstyle.css

storage > logs > log.txt

```
347 [2024-10-03 16:30:00] local.INFO: [2024-10-03 16:30:00] GET http://127.0.0.1:8000/home
348 [2024-10-03 16:30:00] local.INFO: [2024-10-03 16:30:00] GET http://127.0.0.1:8000/accessDenied
349 [2024-10-03 16:33:07] local.INFO: [2024-10-03 16:33:07] GET http://127.0.0.1:8000
350 [2024-10-03 16:33:13] local.INFO: [2024-10-03 16:33:13] POST http://127.0.0.1:8000/set-username
351 [2024-10-03 16:33:13] local.INFO: [2024-10-03 16:33:13] GET http://127.0.0.1:8000/home
352 [2024-10-03 16:33:24] local.INFO: [2024-10-03 16:33:24] GET http://127.0.0.1:8000
353 [2024-10-03 16:33:29] local.INFO: [2024-10-03 16:33:29] POST http://127.0.0.1:8000/set-username
354 [2024-10-03 16:33:30] local.INFO: [2024-10-03 16:33:30] GET http://127.0.0.1:8000/home
355 [2024-10-03 16:33:30] local.INFO: [2024-10-03 16:33:30] GET http://127.0.0.1:8000/accessDenied
356 [2024-10-03 16:33:37] local.INFO: [2024-10-03 16:33:37] GET http://127.0.0.1:8000
357 [2024-10-03 16:34:37] local.INFO: [2024-10-03 16:34:37] POST http://127.0.0.1:8000/set-username
358 [2024-10-03 16:34:37] local.INFO: [2024-10-03 16:34:37] GET http://127.0.0.1:8000/home
359 [2024-10-03 16:35:11] local.INFO: [2024-10-03 16:35:11] POST http://127.0.0.1:8000/set-username
360 [2024-10-03 16:35:11] local.INFO: [2024-10-03 16:35:11] GET http://127.0.0.1:8000/home
361 [2024-10-03 16:35:57] local.INFO: [2024-10-03 16:35:57] GET http://127.0.0.1:8000/restricted
362 [2024-10-03 16:35:57] local.INFO: [2024-10-03 16:35:57] GET http://127.0.0.1:8000/accessDenied
363 [2024-10-03 16:36:52] local.INFO: [2024-10-03 16:36:52] GET http://127.0.0.1:8000/restricted
364 [2024-10-03 16:36:53] local.INFO: [2024-10-03 16:36:53] GET http://127.0.0.1:8000/accessDenied
365 [2024-10-03 16:42:17] local.INFO: [2024-10-03 16:42:17] GET http://127.0.0.1:8000
366 [2024-10-03 16:43:34] local.INFO: [2024-10-03 16:43:34] POST http://127.0.0.1:8000/set-username
367 [2024-10-03 16:43:34] local.INFO: [2024-10-03 16:43:34] GET http://127.0.0.1:8000/home
368 [2024-10-03 16:43:41] local.INFO: [2024-10-03 16:43:41] GET http://127.0.0.1:8000/restricted
```

This is our log.txt file. Here, all log details, including the page URL, timestamp, HTTP method, and any other relevant information, are appended with each request made to the website. This file serves as a record of all incoming traffic, helping us track user activity, request types, and when each action occurred.