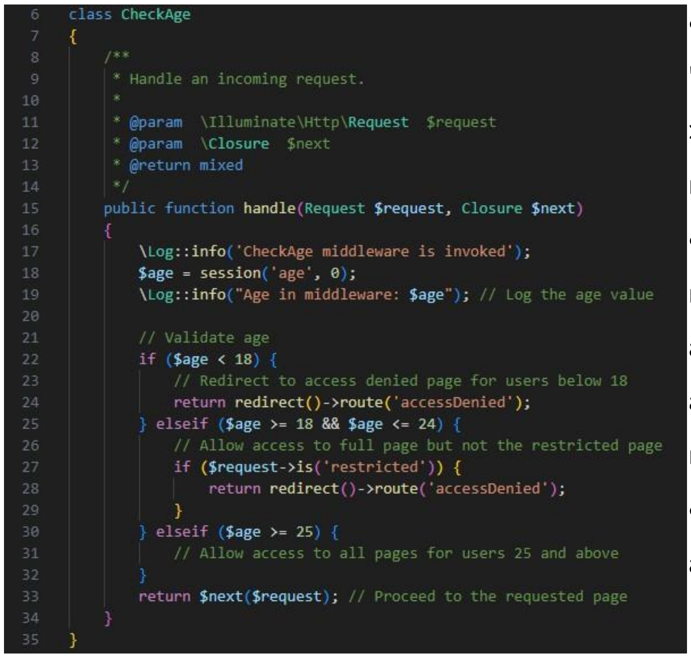
CheckAge Middleware

The CheckAge middleware is designed to control access to different pages of a website based on the user's age, which is retrieved from the session. If no age is stored, it defaults to 0. When a request is made, the middleware's handle method checks the user's age and applies specific access rules:



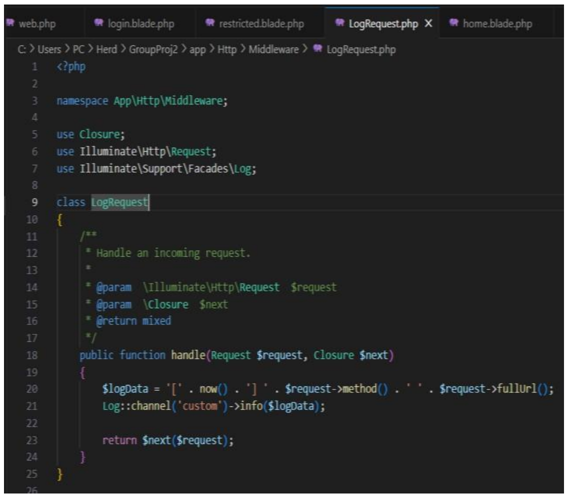
```php
class CheckAge
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        \Log::info('CheckAge middleware is invoked');
        $age = session('age', 0);
        \Log::info("Age in middleware: $age"); // Log the age value

        // Validate age
        if ($age < 18) {
            // Redirect to access denied page for users below 18
            return redirect()->route('accessDenied');
        } elseif ($age >= 18 && $age <= 24) {
            // Allow access to full page but not the restricted page
            if ($request->is('restricted')) {
                return redirect()->route('accessDenied');
            }
        } elseif ($age >= 25) {
            // Allow access to all pages for users 25 and above
        }
        return $next($request); // Proceed to the requested page
    }
}
```

- Under 18: Users are redirected to an "Access Denied" page using redirect() >route('accessDenied'), blocking access to the main content.
- Ages 18 to 24: These users can access most pages but are restricted from specific areas, such as the "Restricted" page. If they attempt to access restricted content, they are redirected to the "Access Denied" page.
- Ages 25 and Above: Users have full access to all pages without restrictions.

In addition to managing access, the middleware logs each instance it is triggered along with the user's age, using \Log::info() for debugging and tracking purposes. After verifying the age, it either redirects the user or allows the request to proceed by calling $next($request). This middleware ensures age-based access control, secures sensitive content, and provides valuable logs for tracking unauthorized access attempts.

LogRequest Middleware

The LogRequest middleware in Laravel logs details of each incoming HTTP request, helping to monitor user interactions and support debugging. When a request is made to the website, this middleware captures key information such as the date and time, the HTTP method (e.g., GET or POST), and the full URL.Namespace and Imports: Defined in App\Http\Middleware, it imports necessary classes, including Closure, Request, and Laravel's Log facade.

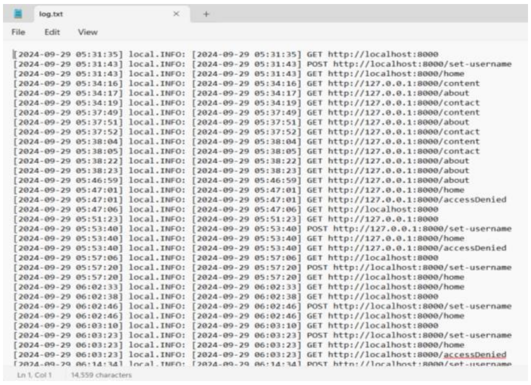

web.php
login.blade.php
restricted.blade.php
LogRequest.php
home.blade.php
C: > Users > PC > Herd > GroupProj2 > app > Http > Middleware > LogRequest.php

```php
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Log;

class LogRequest
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    public function handle(Request $request, Closure $next)
    {
        $logData = '[' . now() . '] ' . $request->method() . ' ' . $request->fullUrl();
        Log::channel('custom')->info($logData);

        return $next($request);
    }
}
```

Handle Method:

The handle method constructs a log message containing the current timestamp, HTTP method, and the URL being accessed.This information is logged at the informational level using a custom logging channel.

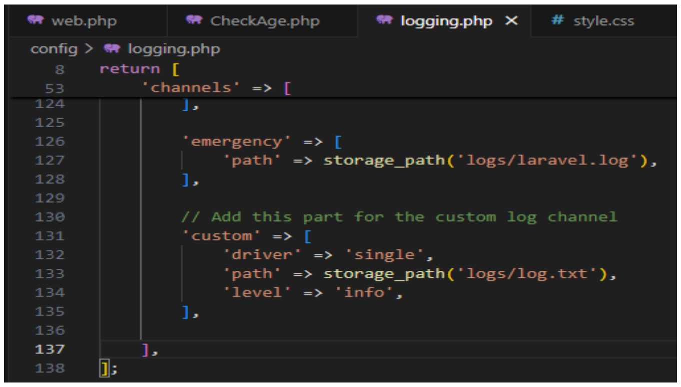
Request Lifecycle: After logging, the middleware calls $next($request), allowing the request to continue through the rest of the middleware stack or reach the intended route. This setup ensures a record of all requests, stored in a custom log file, which provides valuable insights for tracking and analyzing site activity and debugging application performance.



```
[2024-09-29 05:31:35] local.INFO: [2024-09-29 05:31:35] GET http://localhost:8000
[2024-09-29 05:31:43] local.INFO: [2024-09-29 05:31:43] POST http://localhost:8000/set-username
[2024-09-29 05:31:43] local.INFO: [2024-09-29 05:31:43] GET http://localhost:8000/home
[2024-09-29 05:34:16] local.INFO: [2024-09-29 05:34:16] GET http://127.0.0.1:8000/content
[2024-09-29 05:34:17] local.INFO: [2024-09-29 05:34:17] GET http://127.0.0.1:8000/about
[2024-09-29 05:34:19] local.INFO: [2024-09-29 05:34:19] GET http://127.0.0.1:8000/contact
[2024-09-29 05:37:49] local.INFO: [2024-09-29 05:37:49] GET http://127.0.0.1:8000/content
[2024-09-29 05:37:51] local.INFO: [2024-09-29 05:37:51] GET http://127.0.0.1:8000/about
[2024-09-29 05:37:52] local.INFO: [2024-09-29 05:37:52] GET http://127.0.0.1:8000/contact
[2024-09-29 05:38:04] local.INFO: [2024-09-29 05:38:04] GET http://127.0.0.1:8000/content
[2024-09-29 05:38:05] local.INFO: [2024-09-29 05:38:05] GET http://127.0.0.1:8000/contact
[2024-09-29 05:38:22] local.INFO: [2024-09-29 05:38:22] GET http://127.0.0.1:8000/about
[2024-09-29 05:38:23] local.INFO: [2024-09-29 05:38:23] GET http://127.0.0.1:8000/about
[2024-09-29 05:46:59] local.INFO: [2024-09-29 05:46:59] GET http://127.0.0.1:8000/about
[2024-09-29 05:47:01] local.INFO: [2024-09-29 05:47:01] GET http://127.0.0.1:8000/home
[2024-09-29 05:47:01] local.INFO: [2024-09-29 05:47:01] GET http://127.0.0.1:8000/accessDenied
[2024-09-29 05:47:06] local.INFO: [2024-09-29 05:47:06] GET http://localhost:8000
[2024-09-29 05:51:23] local.INFO: [2024-09-29 05:51:23] GET http://127.0.0.1:8000
[2024-09-29 05:53:40] local.INFO: [2024-09-29 05:53:40] POST http://127.0.0.1:8000/set-username
[2024-09-29 05:53:40] local.INFO: [2024-09-29 05:53:40] GET http://127.0.0.1:8000/home
[2024-09-29 05:53:40] local.INFO: [2024-09-29 05:53:40] GET http://127.0.0.1:8000/accessDenied
[2024-09-29 05:57:06] local.INFO: [2024-09-29 05:57:06] GET http://localhost:8000
[2024-09-29 05:57:20] local.INFO: [2024-09-29 05:57:20] POST http://localhost:8000/set-username
[2024-09-29 05:57:20] local.INFO: [2024-09-29 05:57:20] GET http://localhost:8000/home
[2024-09-29 06:02:33] local.INFO: [2024-09-29 06:02:33] GET http://localhost:8000/home
[2024-09-29 06:02:38] local.INFO: [2024-09-29 06:02:38] GET http://localhost:8000
[2024-09-29 06:02:46] local.INFO: [2024-09-29 06:02:46] POST http://localhost:8000/set-username
[2024-09-29 06:02:46] local.INFO: [2024-09-29 06:02:46] GET http://localhost:8000/home
[2024-09-29 06:03:10] local.INFO: [2024-09-29 06:03:10] GET http://localhost:8000
[2024-09-29 06:03:23] local.INFO: [2024-09-29 06:03:23] POST http://localhost:8000/set-username
[2024-09-29 06:03:23] local.INFO: [2024-09-29 06:03:23] GET http://localhost:8000/home
[2024-09-29 06:03:23] local.INFO: [2024-09-29 06:03:23] GET http://localhost:8000/accessDenied
[2024-09-29 06:14:34] local.INFO: [2024-09-29 06:14:34] POST http://localhost:8000/set-username
```

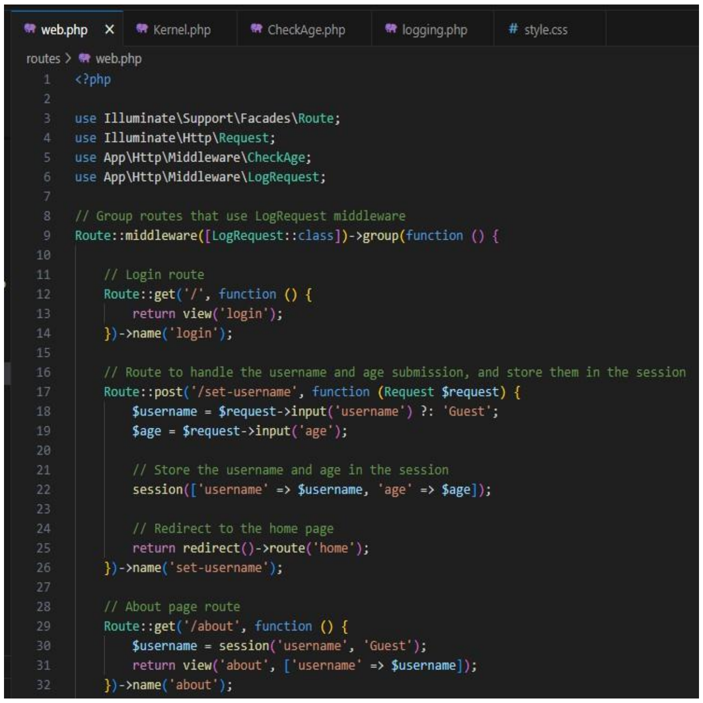
Config/logging.php

To enable the LogRequest middleware to save log data in log.txt, a custom log channel is created in the config/logging.php file. This channel is configured with a 'single' driver, which directs all log entries from the LogRequest middleware to a single file, log.txt, located in the storage/logs folder. The logging.php file in Laravel allows developers to customize how log messages are handled, including setting up various logging channels, specifying logging levels, and defining storage locations. This configuration aids in debugging, monitoring application performance, and auditing user actions by ensuring that essential details like URLs and timestamps are consistently stored in a dedicated log file.



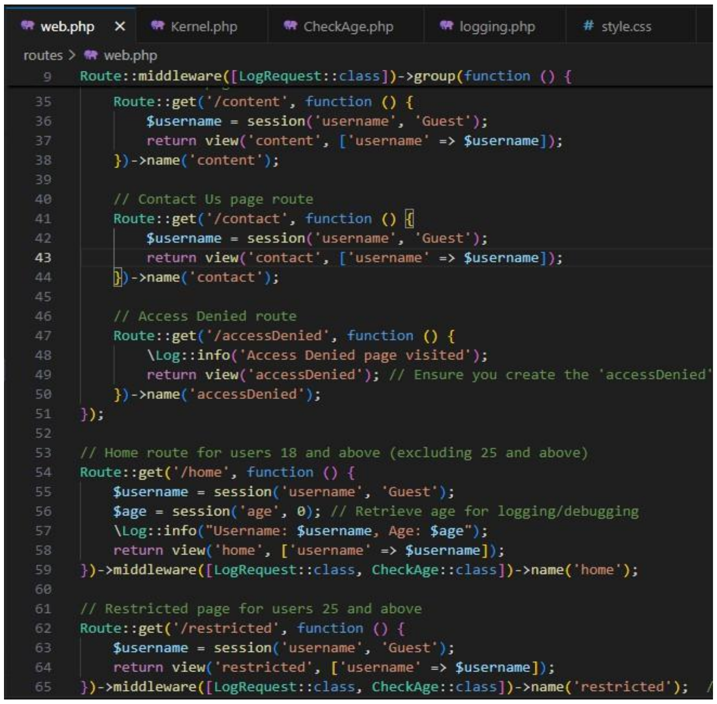
```
return [
    'channels' => [
        ],

        'emergency' => [
            'path' => storage_path('logs/laravel.log'),
        ],

        // Add this part for the custom log channel
        'custom' => [
            'driver' => 'single',
            'path' => storage_path('logs/log.txt'),
            'level' => 'info',
        ],

    ],
];
```

This routing configuration defines how users can access various pages in the application. Routes are grouped under the LogRequest middleware, which logs details about each request. The home page (/) displays the login view where users can enter their username and age, while the /set-username route processes the submitted username and age, stores them in the session, and redirects users to the home page (/home). Additional routes for the About, Content, and Contact pages retrieve the username from the session and display their respective views. An Access Denied route (/accessDenied) logs each visit and shows the "Access Denied" view to users who do not meet age requirements. The Home route (/home) is protected by both LogRequest and CheckAge middlewares, ensuring that only users aged 18 and older can access it. This route retrieves the username and age from the session, logs this information, and displays the home view, while users under 18 are redirected to the Access Denied route. Similarly, the Restricted Page route (/restricted) is accessible only to users aged 25 and older and retrieves the username to display the restricted view.


web.php × | Kernel.php | CheckAge.php | logging.php | # style.css

routes > web.php

```php
<?php

use Illuminate\Support\Facades\Route;
use Illuminate\Http\Request;
use App\Http\Middleware\CheckAge;
use App\Http\Middleware\LogRequest;

// Group routes that use LogRequest middleware
Route::middleware([LogRequest::class])->group(function () {

    // Login route
    Route::get('/', function () {
        return view('login');
    })->name('login');

    // Route to handle the username and age submission, and store them in the session
    Route::post('/set-username', function (Request $request) {
        $username = $request->input('username') ?: 'Guest';
        $age = $request->input('age');

        // Store the username and age in the session
        session(['username' => $username, 'age' => $age]);

        // Redirect to the home page
        return redirect()->route('home');
    })->name('set-username');

    // About page route
    Route::get('/about', function () {
        $username = session('username', 'Guest');
        return view('about', ['username' => $username]);
    })->name('about');
```


web.php × | Kernel.php | CheckAge.php | logging.php | # style.css

routes > web.php

```php
Route::middleware([LogRequest::class])->group(function () {
    Route::get('/content', function () {
        $username = session('username', 'Guest');
        return view('content', ['username' => $username]);
    })->name('content');

    // Contact Us page route
    Route::get('/contact', function () {
        $username = session('username', 'Guest');
        return view('contact', ['username' => $username]);
    })->name('contact');

    // Access Denied route
    Route::get('/accessDenied', function () {
        \Log::info('Access Denied page visited');
        return view('accessDenied'); // Ensure you create the 'accessDenied'
    })->name('accessDenied');
});

// Home route for users 18 and above (excluding 25 and above)
Route::get('/home', function () {
    $username = session('username', 'Guest');
    $age = session('age', 0); // Retrieve age for logging/debugging
    \Log::info("Username: $username, Age: $age");
    return view('home', ['username' => $username]);
})->middleware([LogRequest::class, CheckAge::class])->name('home');

// Restricted page for users 25 and above
Route::get('/restricted', function () {
    $username = session('username', 'Guest');
    return view('restricted', ['username' => $username]);
})->middleware([LogRequest::class, CheckAge::class])->name('restricted');  /
```