MIDDLEWARE

1. Applies middleware to a group of routes.

```php
// Group routes that use the 'web' middleware and Log requests
Route::middleware(['web', LogRequests::class])->group(function () {
```

   a. LogRequests is a custom middleware that contains logic to record the request made by the user, and the info will be appended to the storage/log/logs.txt file.

```
[2024-10-03 16:39:25] local.INFO: [2024-10-03 16:39:25] POST http://localhost:8000/homepage
[2024-10-03 16:39:26] local.INFO: [2024-10-03 16:39:26] GET http://localhost:8000/homepage/Karylle
[2024-10-03 16:39:36] local.INFO: [2024-10-03 16:39:36] POST http://localhost:8000/homepage
[2024-10-03 16:39:36] local.INFO: [2024-10-03 16:39:36] GET http://localhost:8000/restrictedArea/Karylle
[2024-10-03 16:39:45] local.INFO: [2024-10-03 16:39:45] GET http://localhost:8000/menu/Karylle
[2024-10-03 16:39:47] local.INFO: [2024-10-03 16:39:47] GET http://localhost:8000/aboutus/Karylle
[2024-10-03 16:39:49] local.INFO: [2024-10-03 16:39:49] GET http://localhost:8000/contactus/Karylle
[2024-10-03 16:39:51] local.INFO: [2024-10-03 16:39:51] GET http://localhost:8000/homepage/Karylle

```

   b. Grouping routes: The group() method groups several routes together, so that all the routes within this group inherit the middleware.

2. Jaylo_PrinceLouis_Lab1-main > app > Http > Kernel.php

```php
use Illuminate\Foundation\Http\Kernel as HttpKernel;

Codeium: Refactor | Explain
class Kernel extends HttpKernel
{
    protected $middleware = [
        // Global HTTP middleware
        \App\Http\Middleware\LogRequests::class, // Register LogRequests as global middleware
    ];

    protected $middlewareGroups = [
        'web' => [
            \App\Http\Middleware\EncryptCookies::class,
            \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
            \Illuminate\Session\Middleware\StartSession::class,
            \App\Http\Middleware\CheckAge::class,

        ],

        'api' => [
            'throttle:api',
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],
    ];

    protected $routeMiddleware = [
        'check.age' => \App\Http\Middleware\CheckAge::class,
    ];
```

This code defines the HTTP kernel in a Laravel application, registering global middleware like LogRequests and grouping middleware under web and api. It also registers route-specific middleware like CheckAge, ensuring middleware is applied too the appropriate requests.

a.

```php
Codeium: Refactor | Explain
class LogRequests
{
    /**
     * Handle an incoming request.
     *
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)  $next
     */
    Codeium: Refactor | Explain | ×
    public function handle(Request $request, Closure $next): Response
    {
        Log::info('LogRequests middleware executed'); // Log a simple message
        $logData = '[' . now() . '] ' . $request->method() . ' ' . $request->fullUrl();
        Log::channel('custom')->info($logData);

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

This code defines a middleware class called *LogRequests* in a Laravel application. It logs each request with a string containing the current timestamp, request method (GET, POST, etc.), and full URL. This log data is sent to a custom logging channel named "custom." The actual file location for the logs depends on how the custom channel is configured in the application's logging configuration file *(config/logging.php)*.

config/logging.php

```php
        'custom' => [
            'driver' => 'single',
            'path' => storage_path('logs/log.txt'),
            'level' => 'info',
        ],

    ],
```

storage/logs/log.txt

```
Jaylo_PrinceLouis_Lab1-main > storage > logs > log.txt
365  [2024-10-05 13:28:42] local.INFO: [2024-10-05 13:28:42] POST http://localhost:8000/homepage
366  [2024-10-05 13:28:42] local.INFO: [2024-10-05 13:28:42] GET http://localhost:8000/accessDenied
367  [2024-10-05 13:28:45] local.INFO: [2024-10-05 13:28:45] GET http://localhost:8000/goTologin
368  [2024-10-05 13:29:06] local.INFO: [2024-10-05 13:29:06] POST http://localhost:8000/homepage
369  [2024-10-05 13:29:06] local.INFO: [2024-10-05 13:29:06] GET http://localhost:8000/homepage/Karila
370  [2024-10-05 13:29:17] local.INFO: [2024-10-05 13:29:17] POST http://localhost:8000/homepage
371  [2024-10-05 13:29:17] local.INFO: [2024-10-05 13:29:17] GET http://localhost:8000/restrictedArea/Karila
372  [2024-10-05 13:29:25] local.INFO: [2024-10-05 13:29:25] GET http://localhost:8000/contactus/Karila
373  [2024-10-05 13:29:27] local.INFO: [2024-10-05 13:29:27] GET http://localhost:8000/aboutus/Karila
374  [2024-10-05 13:29:31] local.INFO: [2024-10-05 13:29:31] GET http://localhost:8000/menu/Karila
375  [2024-10-05 13:29:35] local.INFO: [2024-10-05 13:29:35] GET http://localhost:8000/homepage/Karila
376
```

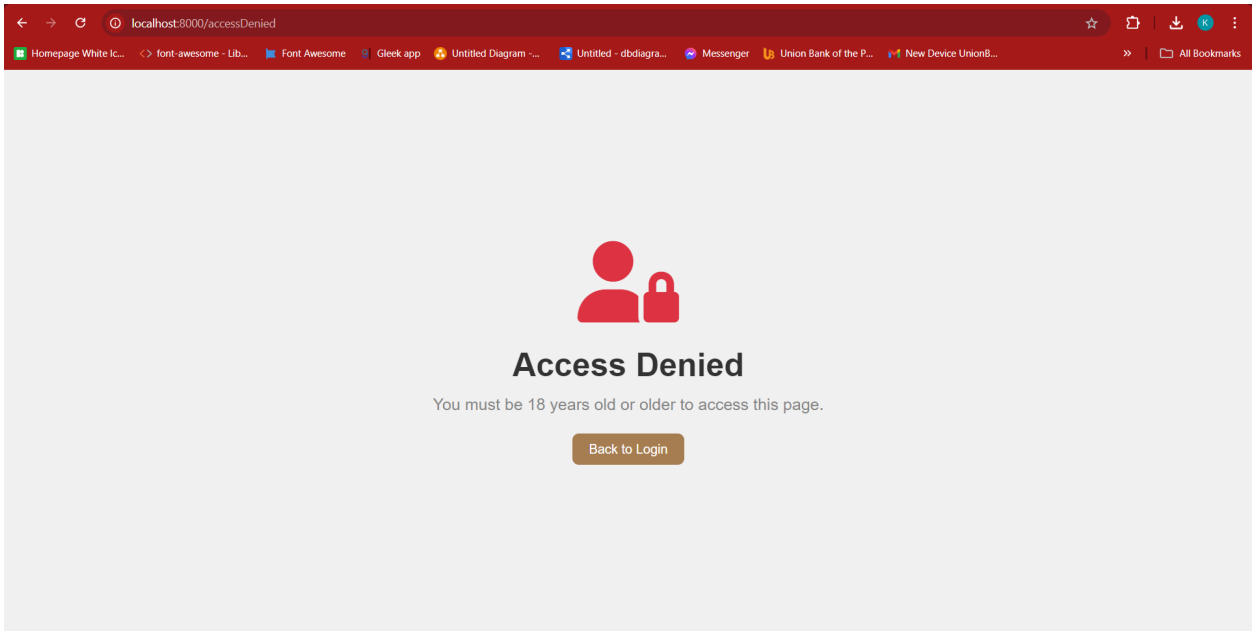
b.

Jaylo_PrinceLouis_Lab1-main > app > Http > Middleware > CheckAge.php

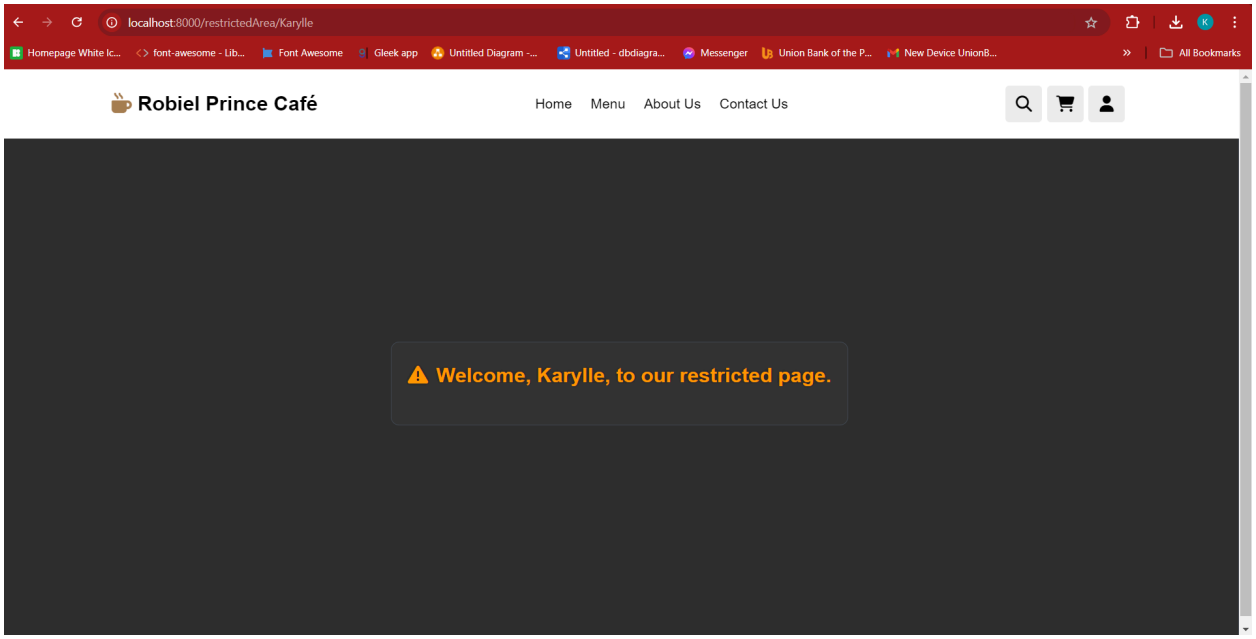
```
10  {
19      {
39
40
41          // Check if the user is under 18
42          if ($age < 18) {
43              // Redirect to Access Denied page if under 18
44              return redirect('/accessDenied')->with('error', 'You must be 18 or older to access this page.');
45          } elseif ($age >= 18 && $age <= 20) {
46              // Redirect to homepage if age is between 18 and 20
47              return redirect("/homepage/" . $username);
48          } elseif ($age >= 21) {
49
50              \Log::info('Redirecting to restricted area for user: ' . $username);
51              // Redirect to restricted area if 21 or older
52              return redirect("/restrictedArea/" . $username);
53          }
54
55          // Proceed if 18 or older
56          return $next($request);
57      }
58  }
```

This is the *CheckAge* middlewarethat checks a user's age based on their birthdate input. It validates the format, logs the user's age and username, and redirects them: under 18 to an "*Access Denied*" page, 18 to 20 to the homepage, and 21 or older to a restricted area.

Access Denied page



Restricted page for 21 and above



3. Middleware in routes

```php
// Route for restricted area with dynamic username
Route::get('/restrictedArea/{username}', function ($username) {
    return view('restrictedArea', ['username' => $username]);
})->name('restrictedArea');

// Middleware for age verification, restricted access, or homepage access post-login
Route::post('/homepage', function (Request $request) {
    return view('homepage', ['username' => $request->input('username')]);
})->middleware(CheckAge::class);

// Access Denied route
Route::get('/accessDenied', function () {
    return view('accessDenied');
})->name('accessDenied');
```

The *Route::get* route sets up a route for */restrictedArea/{username}*, which dynamically handles the username and returns a view named restrictedArea, passing the username to it.

The *Route::post* route for /homepage applies the *CheckAge* middleware. After the middleware validates the user's age, it returns a view named homepage, also passing the username to it.

The *CheckAge* middleware filters the age for all routes it's applied to, including the *restrictedArea* page. If a user is under 21, they won't be redirected there. The middleware ensures only users who meet the age requirements access specific sections of the site.

4. Test if middleware if working properly