

# Laravel - Session

Topics : [Laravel](#)

Written on [December 25, 2023](#)

In Laravel, the session handling mechanism is quite robust and provides a convenient way to store information across multiple requests. Sessions can be used to store user data, flash messages, and other information that needs to persist between requests. Here's a guide on working with sessions in Laravel:

## 1. Configuration:

Ensure that session configuration is set up correctly in the `config/session.php` file. The default configuration should work for most applications.

## 2. Starting a Session:

Sessions are started automatically by Laravel, but you can manually start a session using the session helper or the Session facade:

```
session_start(); // using the session helper

// or

use Illuminate\Support\Facades\Session;

Session::start(); // using the Session facade
```

## 3. Storing Data in the Session:

You can store data in the session using the `put` method or the session helper:

```
// using the session helper
session(['key' => 'value']);

// using the Session facade
Session::put('key', 'value');
```

## 4. Retrieving Data from the Session:

To retrieve data from the session, you can use the `get` method or the session helper:

```
// using the session helper
$value = session('key', 'default');
```

```
// using the Session facade
$value = Session::get('key', 'default');
```

## 5. Removing Data from the Session:

You can remove data from the session using the `forget` method or the `session` helper:

```
// using the session helper
session()->forget('key');
```

```
// using the Session facade
Session::forget('key');
```

## 6. Flashing Data:

Flash data is only stored in the session for the next request and then is automatically forgotten. This is useful for temporary messages like success messages after a form submission:

```
// using the session helper
session()->flash('key', 'value');
```

```
// using the Session facade
Session::flash('key', 'value');
```

## 7. Retrieving All Data:

To retrieve all data from the session, you can use the `all` method:

```
// using the session helper
$data = session()->all();
```

```
// using the Session facade
$data = Session::all();
```

## 8. Checking if a Key Exists:

You can check if a key exists in the session using the `has` method:

```
// using the session helper
if (session()->has('key')) {
    // Key exists
}
```

```
// using the Session facade
if (Session::has('key')) {
    // Key exists
}
```

```
}
```

## 9. Regenerating the Session ID:

To regenerate the session ID and prevent session fixation attacks, you can use the regenerate method:

```
// using the session helper  
session()->regenerate();
```

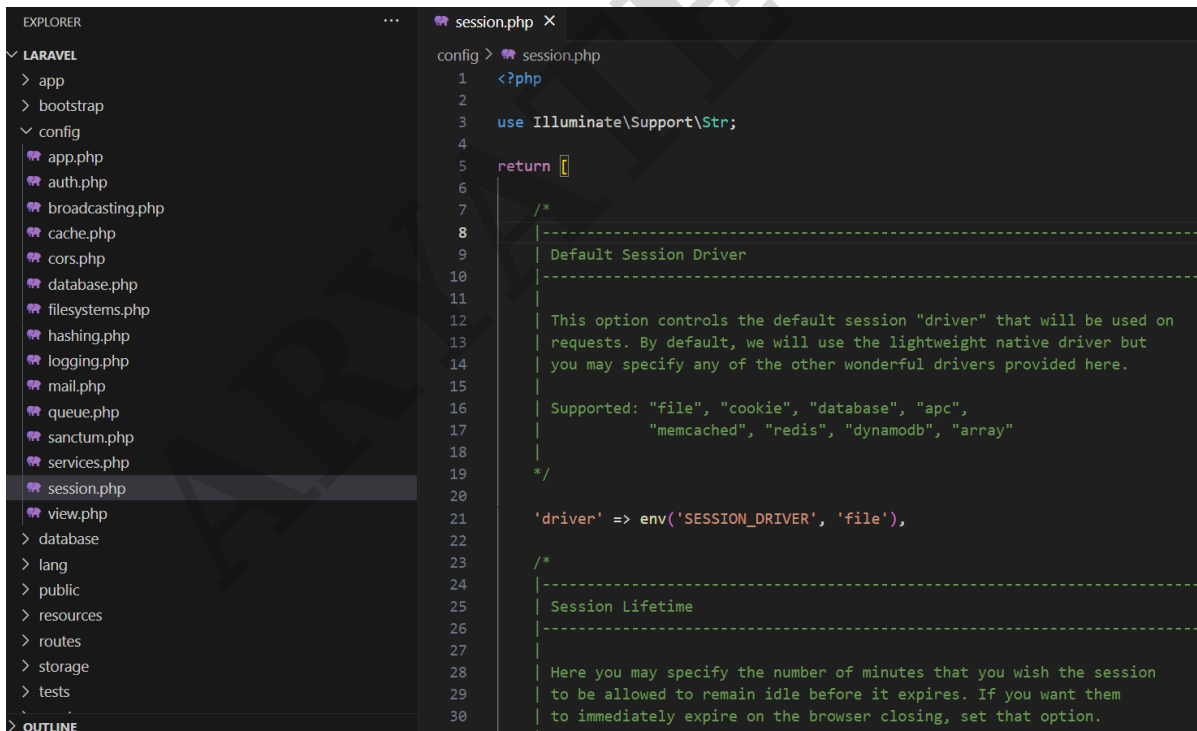
```
// using the Session facade  
Session::regenerate();
```

## 10. Deleting the Session:

To delete the entire session data, you can use the flush method:

```
// using the session helper  
session()->flush();
```

```
// using the Session facade  
Session::flush();
```



```
EXPLORER  
LARAVEL  
  > app  
  > bootstrap  
  > config  
    app.php  
    auth.php  
    broadcasting.php  
    cache.php  
    cors.php  
    database.php  
    filesystems.php  
    hashing.php  
    logging.php  
    mail.php  
    queue.php  
    sanctum.php  
    services.php  
    session.php  
    view.php  
  > database  
  > lang  
  > public  
  > resources  
  > routes  
  > storage  
  > tests  
  > .env  
  > .htaccess  
  > .htpasswd  
  > .user.ini  
  > OUTLINE  
session.php X  
config > session.php  
1  <?php  
2  
3  use Illuminate\Support\Str;  
4  
5  return []  
6  
7  
8  /*  
9  |-----  
10 | Default Session Driver  
11 |-----  
12 | This option controls the default session "driver" that will be used on  
13 | requests. By default, we will use the lightweight native driver but  
14 | you may specify any of the other wonderful drivers provided here.  
15 |  
16 | Supported: "file", "cookie", "database", "apc",  
17 |             "memcached", "redis", "dynamodb", "array"  
18 |-----  
19 |*/  
20  
21  'driver' => env('SESSION_DRIVER', 'file'),  
22  
23  /*  
24 |-----  
25 | Session Lifetime  
26 |-----  
27 | Here you may specify the number of minutes that you wish the session  
28 | to be allowed to remain idle before it expires. If you want them  
29 | to immediately expire on the browser closing, set that option.  
30 |-----  
31 |*/
```