Some termonology

.CSRF TOKEN

1 - Laravel automatically generates a CSRF "token" for each active user session managed by the application. This token is used to verify that the authenticated user is the one actually making the requests to the application.

@csrf

{{csrf\_field()}}

Laravel makes it easy to protect your application from [cross-site request forgery](https://en.wikipedia.org/wiki/Cross-site_request_forgery) (CSRF) attacks. Cross-site request forgeries are a type of malicious exploit whereby unauthorized commands are performed on behalf of an authenticated user.

/

### Storing Data

To store data in the session, you will typically use the put method or the session helper:

// Via a request instance...$request->session()->put('key', 'value');

### Flash Data

Sometimes you may wish to store items in the session only for the next request. You may do so using the flash method. Data stored in the session using this method will only be available during the subsequent HTTP request, and then will be deleted. Flash data is primarily useful for short-lived status messages:

$request->session()->flash('status', 'Task was successful!');

Of course, the GET route will display a form for the user to create a new blog post, while the POST route will store the new blog post in the database.

#### **Stopping On First Validation Failure**

Sometimes you may wish to stop running validation rules on an attribute after the first validation failure. To do so, assign the bail rule to the attribute:

$request->validate([

'title' => 'bail|required|unique:posts|max:255',

'body' => 'required',]);

### Displaying The Validation Errors

<h1>Create Post</h1>

@if ($errors->any())

<div class="alert alert-danger">

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<!-- Create Post Form -->