ScopeTreasure Courses

LARAVEL COURSES · LESSON LV.3 · CODE SCOPE SERIES

Laravel Views + Blade Templates

Pages with brains: echo with {{ }}, @if and @foreach, one master layout with @extends and @yield, and partials with @include.

Introduction

A controller's answer is a **view** — an HTML page with superpowers. Laravel's view engine is called **Blade**: you write normal HTML in files ending `.blade.php`, and Blade lets you print data, run loops and make decisions right inside the page. Before the page reaches the browser, Laravel compiles your Blade into plain PHP and then into honest HTML — the visitor never sees a single Blade symbol.

The rule that keeps everything clean: **the view shows, the controller decides.** A view receives data ready-made and presents it beautifully; it never fetches from the database itself. In this lesson you will print data safely, loop a class list onto the screen, and build one master layout that every page inherits — so your header and footer exist exactly once.

(Roman Urdu: Blade HTML ka samajhdar bhai hai — data chhaap bhi deta hai, loop bhi chalata hai, aur page ka naksha bhi sambhalta hai.)

Key points

- **Blade files live in `resources/views/`** and end in `.blade.php`. The controller opens one with `return view('students.index', [...])` — the dot means "the index file inside the students folder".
- **`{{ }}` prints data — safely:** Blade escapes whatever you print, turning `<script>` into harmless text. This one habit quietly blocks the most common attack on websites (XSS). `{!! !!}` prints raw HTML — use it only for HTML you fully trust, which in practice means almost never.
- **Directives start with `@`:** `@if`, `@foreach`, `@isset` are instructions to Blade, written inside the HTML where they belong. They compile away — view-source in the browser shows pure HTML.
- **Pass data as an array:** the controller sends `['students' => $students]`, and the view uses `$students` like a normal variable. What is not passed does not exist in the view.
- **Layouts kill copy-paste:** a master layout holds the header, nav and footer with `@yield('content')` holes. Each page `@extends` the layout and fills the holes with `@section`. Change the layout once; fifty pages update.

  (Roman Urdu: Layout aik master naksha hai — har page usi mein apna hissa bharta hai, sir saja hua frame dobara nahi banana parta.)
- **`@include` shares small pieces:** a navbar, an alert box, a footer strip — saved once as a partial and included wherever needed: `@include('partials.nav')`.
- **Blade comments are invisible:** `{{-- note --}}` never reaches the browser, unlike HTML comments which anyone can read in view-source.

Code examples

Example 1 — Printing data: the safe echo and the student card

```
// StudentController — the controller decides, the view shows
public function show($id)
{
    $student = Student::findOrFail($id);
    return view('students.show', ['student' => $student]);
}
```

```
{{-- resources/views/students/show.blade.php --}}
<h1>{{ $student->name }}</h1>
<p>City: {{ $student->city }}</p>
<p>Marks: {{ $student->marks }}</p>

@if ($student->marks >= 90)
    <p class="badge">Topper!</p>
@endif
```

What it does: The controller fetches the student (LV.4 explains `findOrFail`) and passes it over. Blade prints each field with `{{ }}` — escaped, always. The `@if` adds a "Topper!" badge only when marks are 90 or more, so Hira (95) and Sara (92) wear it and Bilal (76) does not.

What you will see: for student 4 (Hira Shah): a heading `Hira Shah`, `City: Karachi`, `Marks: 95`, and the Topper badge. For Bilal: the same card, no badge. The browser's view-source shows clean HTML — no `@if`, no `{{ }}` anywhere.

Example 2 — The class list: @foreach builds the table rows

```
<table>
    <tr><th>Name</th><th>Marks</th><th>Result</th></tr>
    @foreach ($students as $student)
        <tr>
            <td>{{ $student->name }}</td>
            <td>{{ $student->marks }}</td>
            <td>{{ $student->marks >= 40 ? 'PASS' : 'FAIL' }}</td>
        </tr>
    @endforeach
</table>
<p>Total students: {{ $students->count() }}</p>
```

What it does: One block of Blade loops over all five students and stamps out one table row each. The ternary `? :` inside the echo prints PASS or FAIL per student. Writing this page took one `<tr>` design — the loop

does the other four.

What you will see: a five-row table — Ali 88 PASS, Sara 92 PASS, Bilal 76 PASS, Hira 95 PASS, Usman 64 PASS — and under it, `Total students: 5`.

Example 3 — One master layout for the whole app

```
{{-- resources/views/layouts/app.blade.php — the master --}}
<!DOCTYPE html>
<html>
<head>
    <title>@yield('title') — School App</title>
</head>
<body>
    @include('partials.nav')

    <main>
        @yield('content')
    </main>

    <footer>School App — built with Laravel</footer>
</body>
</html>
```

```
{{-- resources/views/students/index.blade.php — a child page --}}
@extends('layouts.app')

@section('title', 'All Students')

@section('content')
    <h1>All Students</h1>
    <p>The class list lives here.</p>
@endsection
```

What it does: The layout owns everything repeated: the HTML shell, the nav (included from a partial), the footer, and two holes — `@yield('title')` and `@yield('content')`. The child page declares which layout it extends and fills each hole with a `@section`. Fifty child pages, one frame.

What you will see: the students page wearing the shared nav and footer, with the browser tab titled `All Students — School App`. Edit the layout's footer once and every page in the app shows the new footer instantly.

Quick recap

- Blade = HTML + data printing + decisions: `{{ }}` (safe echo), `@if`, `@foreach`.

- Controllers pass data in; views only present it. Escaping is automatic — `{!! !!}` is the rare, dangerous exception.
- One master layout (`@extends` / `@section` / `@yield`) + partials (`@include`) means every repeated piece exists exactly once.

Common mistakes

1. **Printing raw user data with `{!! !!}`** — if a student saves `<script>` as his name, raw printing runs it in every visitor's browser. Fix: default to `{{ }}` everywhere; treat `{!! !!}` as a loaded tool you almost never pick up.
2. **Fetching data inside the view** — calling `Student::all()` in Blade tangles the layers and hides bugs. Fix: the controller fetches and passes; the view only loops and prints what it was given.
3. **Forgetting `@endsection` / `@endif` / `@endforeach`** — Blade answers with a confusing compile error far from the real line. Fix: write the closing directive the moment you write the opening one, then fill the middle.
4. **Repeating the header in every page** — the copy-paste tax arrives when the header changes. Fix: the first time two pages share a block, it moves into the layout or a partial.

Practice

1. Build the master layout from Example 3 with a nav partial containing three named-route links (LV.2). Make two child pages — students and teachers — and prove that one layout edit appears on both.
2. Take the Example 2 table and add a "Grade" column printed with `@if/@elseif/@else` (A at 90+, B at 75+, C at 60+, F below) instead of the ternary. Check all five students by hand: 95 A, 92 A, 88 B, 76 B, 64 C.

Practice quiz

1. What does {{ }} do to the data it prints, and what attack does that quietly block?

- It prints raw HTML, blocking nothing
- Blade escapes whatever you print, turning <script> into harmless text, which quietly blocks XSS, the most common attack on websites
- It deletes the data after printing
- It only works for numbers

Answer: Blade escapes whatever you print, turning <script> into harmless text, which quietly blocks XSS, the most common attack on websites

2. When may you use {!! !!} instead of {{ }}?

- Always, it is the default
- Almost never: it prints raw HTML, only for HTML you fully trust
- Only for numbers

- Only in layouts

Answer: Almost never: it prints raw HTML, only for HTML you fully trust

3. How do layouts kill copy-paste across pages?

- By copying the header into every page automatically
- A master layout holds header, nav and footer with @yield('content') holes; each page @extends the layout and fills the holes with @section, so changing the layout once updates fifty pages
- Layouts do not help with repetition
- By deleting repeated pages

Answer: A master layout holds header, nav and footer with @yield('content') holes; each page @extends the layout and fills the holes with @section, so changing the layout once updates fifty pages

---

Watch the Code Scope channel: https://www.youtube.com/@CodeScopePK — this lesson's own video joins the channel when it is recorded.

Read online: https://scopetreasure.com/notes/laravel-views-blade · More free tools and notes at https://scopetreasure.com