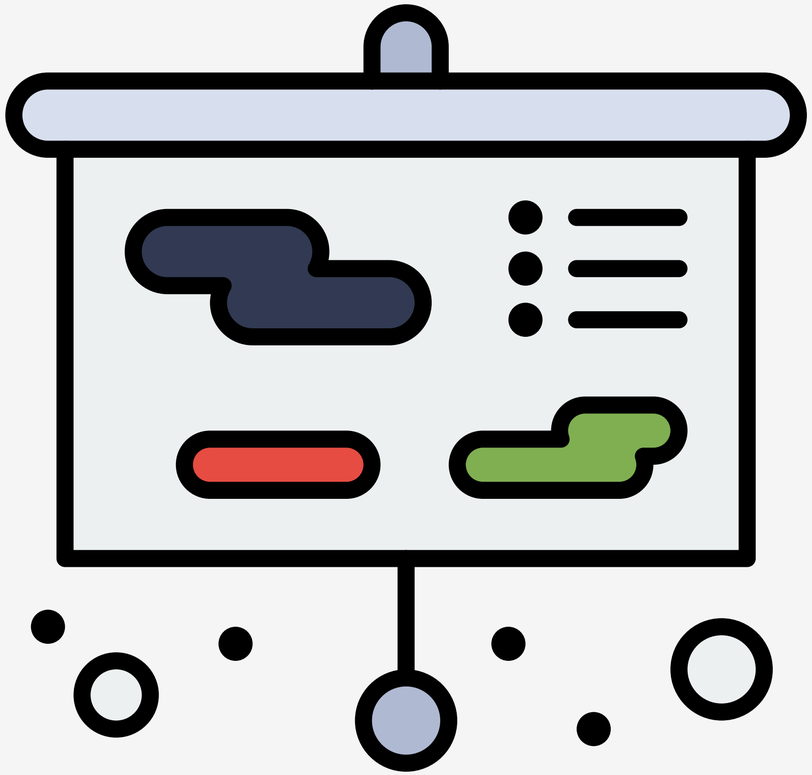
SERVER-SIDE
WEB DEVELOPMENT

Lecture 7

TODAY'S TOPICS

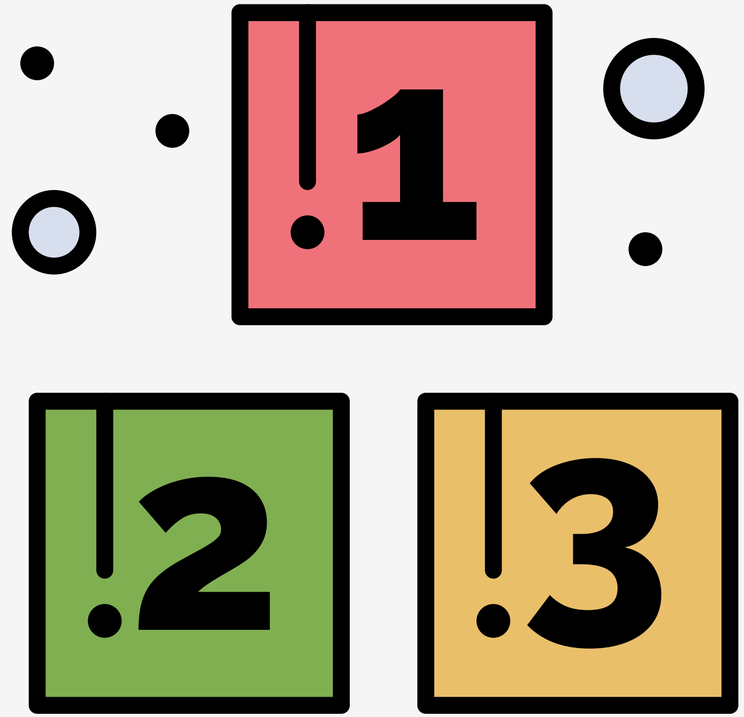


- Introduction to Laravel
- Laravel Routes
- Laravel Views

QUESTIONS

LARAVEL

LARAVEL

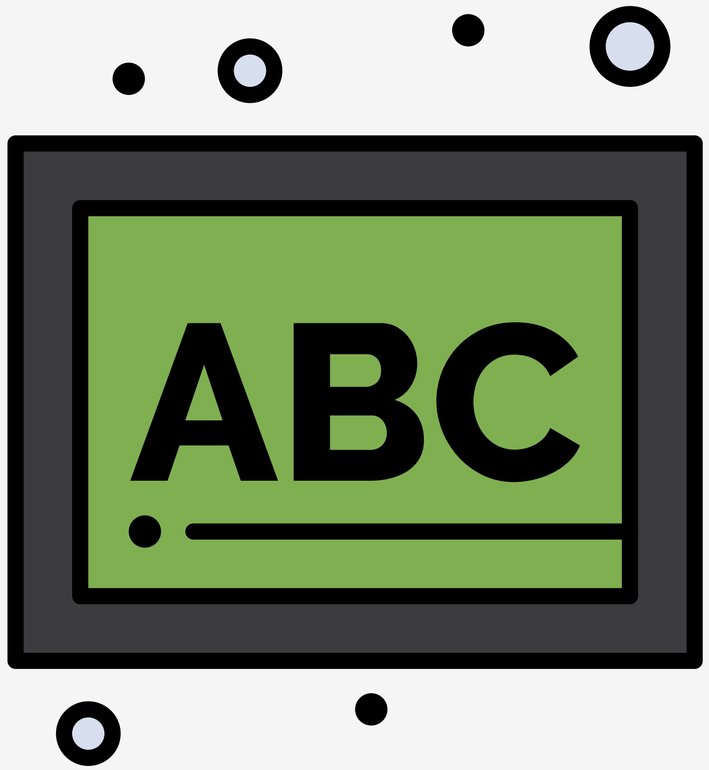


- Free, open-source PHP framework
- Build using OO PHP
- Follows the Model-View-Controller (MVC) architectural pattern
- Supports routing, models, view, controllers, and authentication
- Includes Blade templating engine and Artisan CLI
- Working with Laravel 7.x

INSTALLING LARAVEL

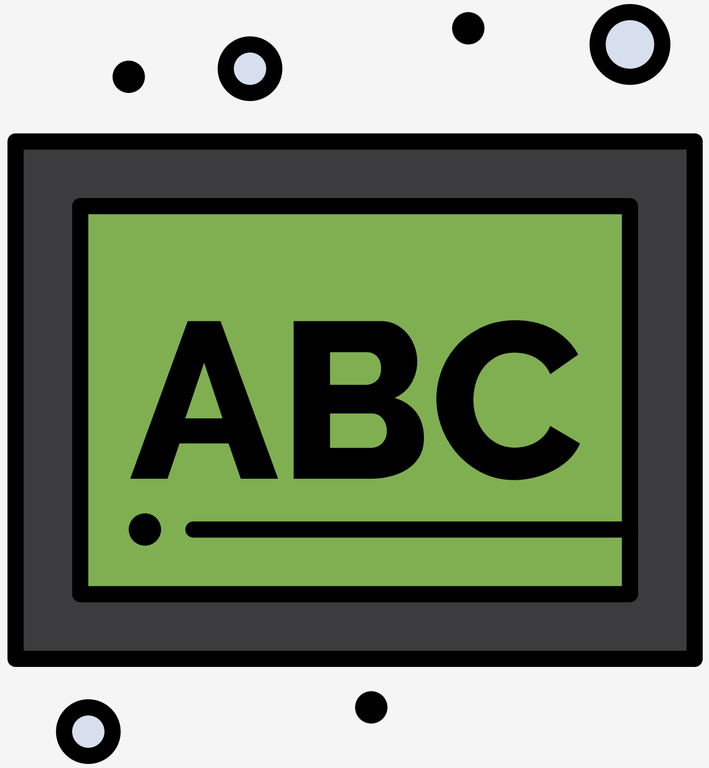
LARAVEL ROUTES

LARAVEL ROUTES



- Routing is one of the fundamental features of Laravel
- Routes create friendly URL scheme that response to HTTP verbs
- Routes are defined in the `routes/web.php` file
- All routes start with a call to the `Route` class and a static method
- Each method take a URI and a function

LARAVEL ROUTES



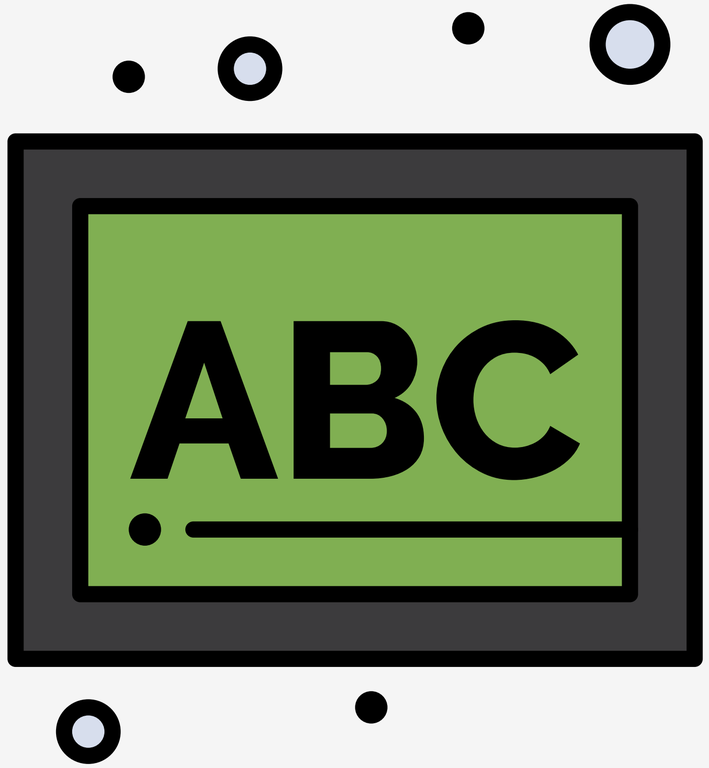
- The URI is the path is enter into the Browser
- The value returned by the function will be displayed in the browser
- The `view` function can be used to return view files
- View files are typically Blade templates and are found in `resources/views`

ROUTES

```php
// routes/web.php
Route::get('/', function () {
    return 'Hello World';
});

Route::get('/welcome', function () {
    return view('welcome');
});
```

LARAVEL ROUTES



- Data can be passed to a view as an associative array
- The associative array is the second argument of the view function
- The data is accessible in the view as variables with the same name as the key.

ROUTES

```php
// routes/web.php
  Route::get('/welcome', function () {
    return view('welcome', ['name' => 'John']);
  });
```

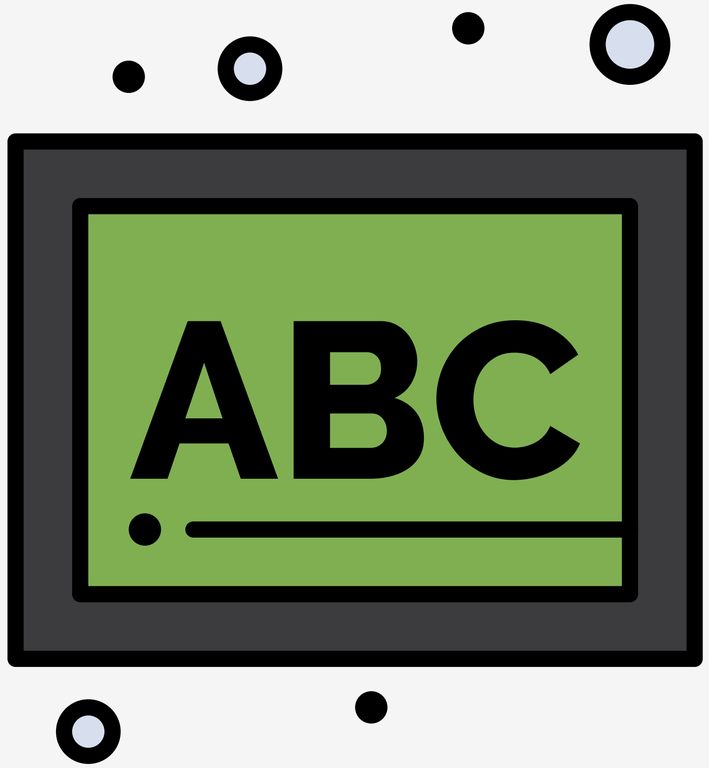
```blade
<!-- resources/views/welcome.blade.php -->
<html>
  <body>
    <h1>
      Hello, <?php echo $name; ?>
    </h1>
  </body>
</html>
```

ROUTES

```php
// routes/web.php
  Route::get('/welcome', function () {
    return view('welcome', ['name' => 'John']);
  });
```

```blade
<!-- resources/views/welcome.blade.php -->
<html>
  <body>
    <h1>
      Hello, {{ $name }}
    </h1>
  </body>
</html>
```

ROUTE PARAMETERS



- Route parameters allows a segment of the URI to be variable
- A route parameter is passed to the route callback function
- Route parameters can be required or optional

ROUTES

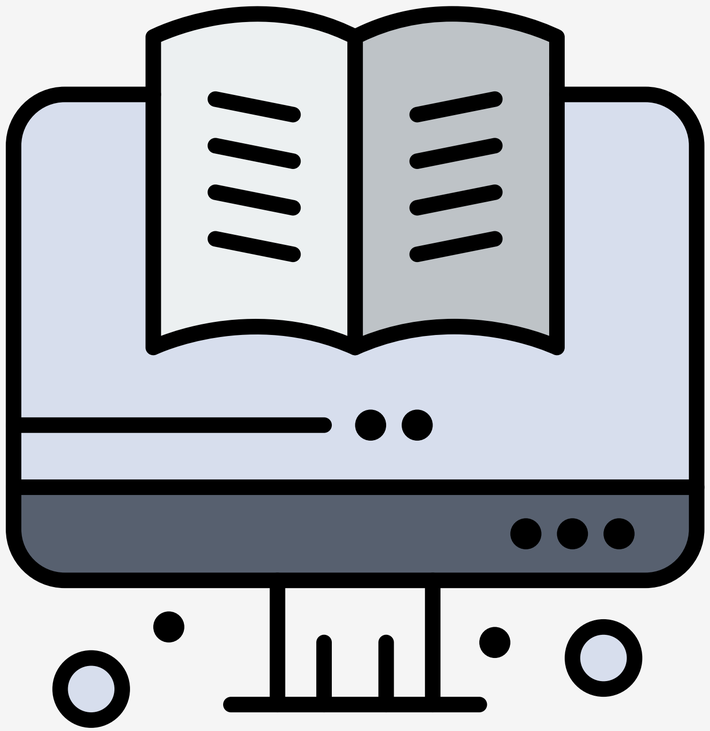
```php
// routes/web.php
Route::get('/posts/{post}', function ($post) {
  $posts = [
    'post-1' => 'This is my first post';
    'post-2' => 'This is my second post';
  ];

  return view('post', [
    'post' => $posts[$post]
  ]);
});
```

HANDS-ON

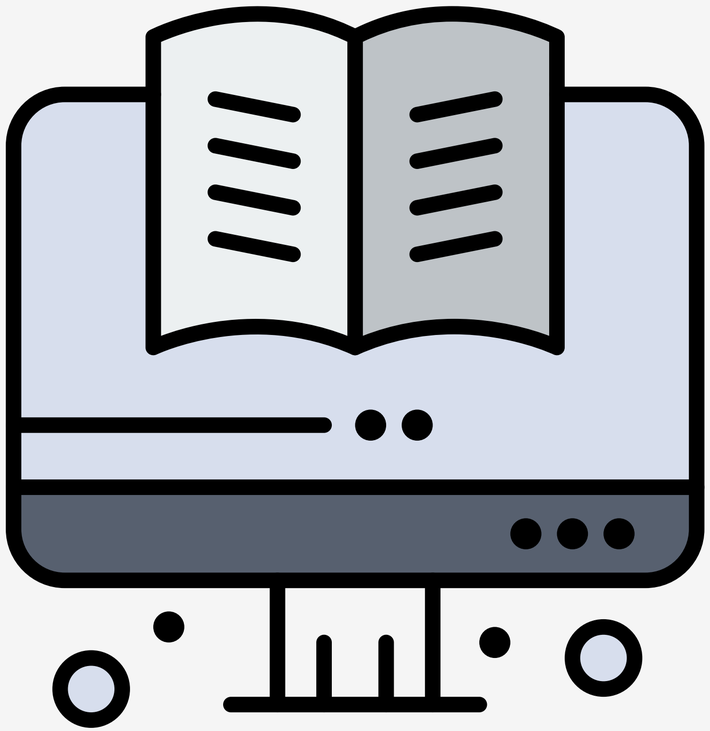
LARAVEL VIEWS

LARAVEL VIEWS



- Laravel follows the Model-View-Controller (MVC) architectural pattern
- MVC provides a separation of task between the data (Model), the interface (View), and the logic (Controller)
- Views are what gets displayed in the browser
- Views are stored in the `resources/views` directory
- Views can PHP or Blade files

LARAVEL BLADE

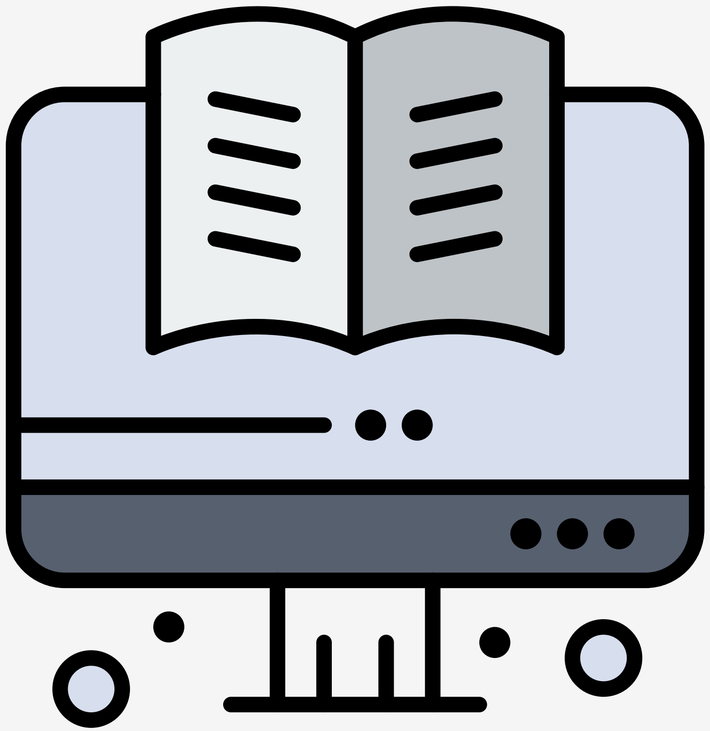


- Blade is a PHP templating engine included with Laravel
- Blade includes multiple statements and directives to simplify creating templates
- Blade can include PHP tags and functions
- Blade is compiled to plain PHP

BLADE

```
<html>
  <body>
    <h1>Hello, {{ $name }}</h1>
  </body>
</html>
```

LARAVEL BLADE



- Blade provides control structures that mimimic their PHP counterparts
- The `@if`, `@elseif`, `@else`, and `@endif` directives can be used for conditional statements
- The `@for` / `@endfor` and `@foreach` / `@endforeach` directives can be use for loops

BLADE

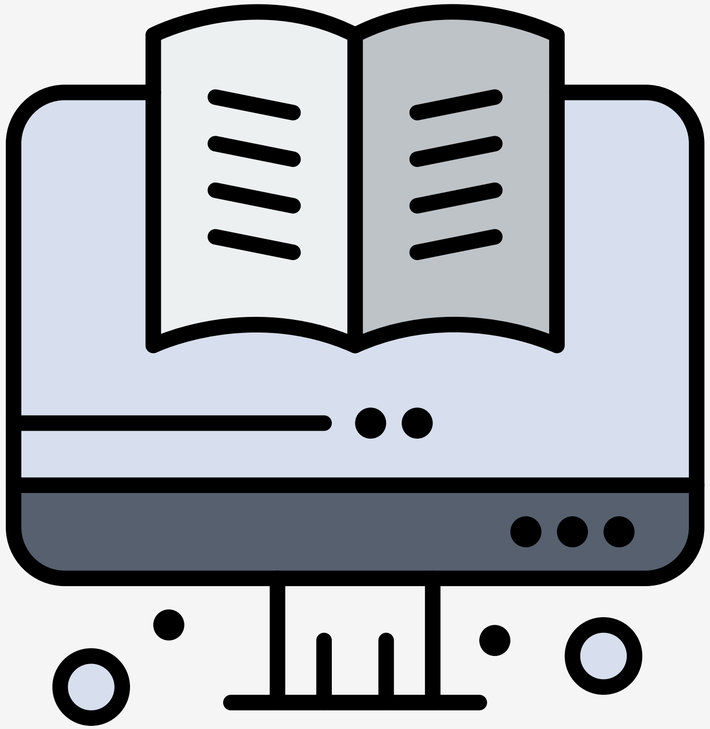
```
<html>
  <body>
    @if (isset($name))
      <h1>Hello, {{ $name }}</h1>
    @else
      <h1>Hello, Guest</h1>
    @endif
  </body>
</html>
```

BLADE

```
<html>
  <body>
    @for ($i = 0; $i < 10; $i++)
      <p>The current value is {{ $i }}</p>
    @endfor

    <ul>
      @foreach ($list as $item)
        <li>$item</li>
      @endforeach
    </ul>
  </body>
</html>
```

LARAVEL BLADE



- Blade provides directives for creating "master" layout pages
- The `@section` directive defined a section of content
- The `@yield` directive display the contents of a given section
- The `@show` directive is used to end a defined section and display the section

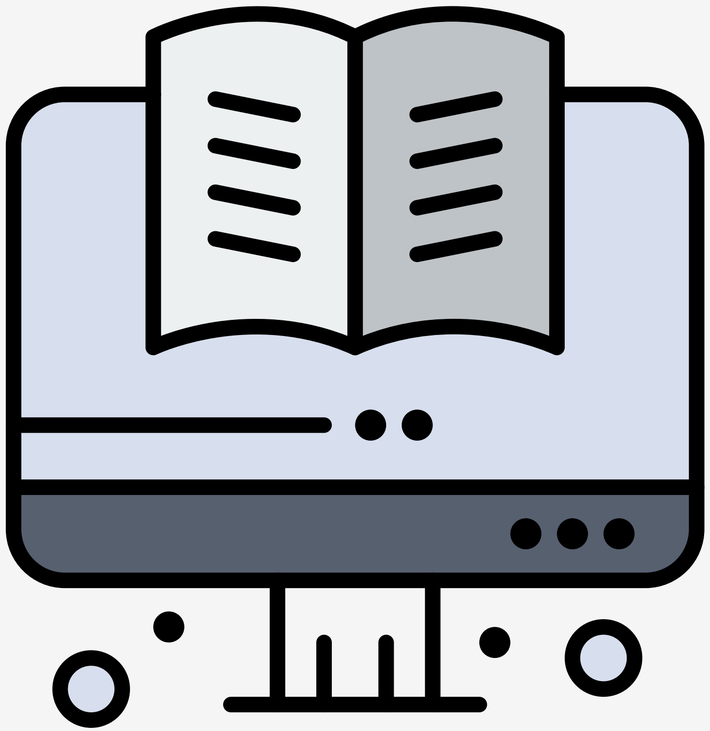
MASTER LAYOUT

```
<!-- resources/views/layouts/app.blade.php -->

<html>
    <head>
        <title>App Name - @yield('title')</title>
    </head>
    <body>
        @section('sidebar')
            This is the master sidebar.
        @show

        <div class="container">
            @yield('content')
        </div>
    </body>
</html>
```

LARAVEL BLADE



- The `@extend` directive specifies which layout should be inherited
- The `@section` directive can be used to inject content
- The `@endsection` directive ends a section without displaying it
- The `@parent` directive includes the parent's content with the child's

CHILD LAYOUT

```
<!-- resources/views/child.blade.php -->

@extends('layouts.app')

@section('title', 'Page Title')

@section('sidebar')
    @parent

    <p>This is appended to the master sidebar.</p>
@endsection

@section('content')
    <p>This is my body content.</p>
@endsection
```

MASTER LAYOUT

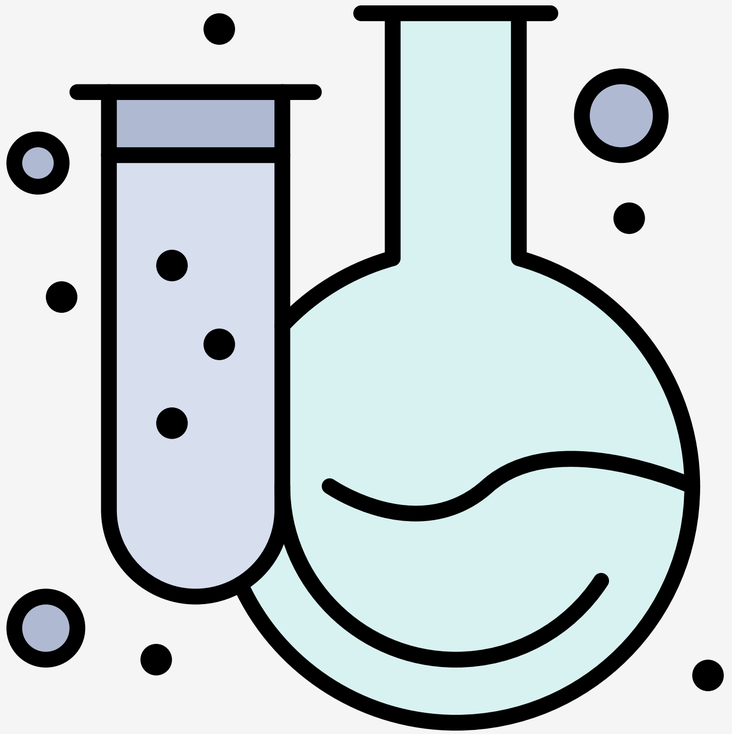
```
<!-- resources/views/layouts/app.blade.php -->

<html>
    <head>
        <title>App Name - @yield('title')</title>
    </head>
    <body>
        @section('sidebar')
            This is the master sidebar.
        @show

        <div class="container">
            @yield('content')
        </div>
    </body>
</html>
```

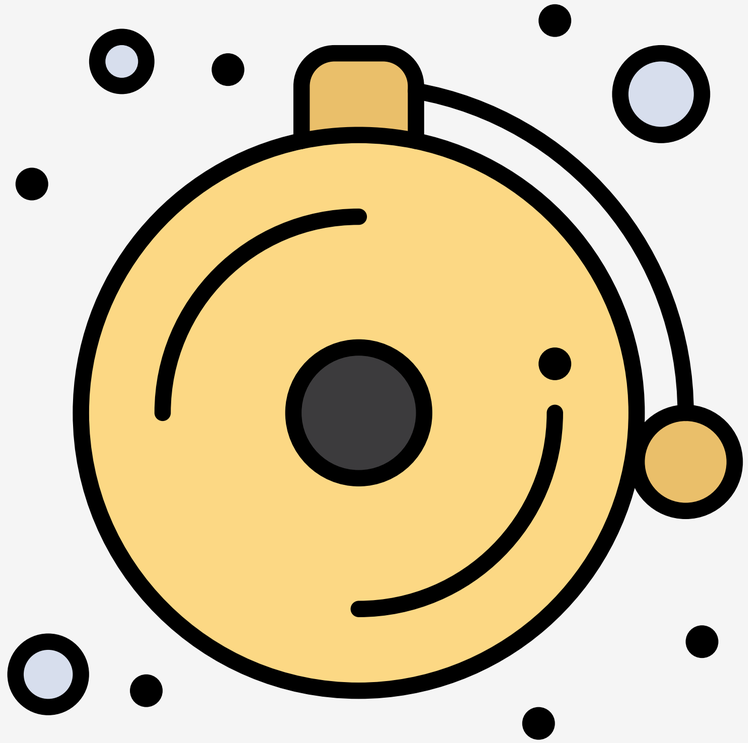
HANDS-ON

HYBRID #4



- Watch the entire course of Laravel 5 Essential Training
- Write 3 to 4 sentences
- *DUE:* **Wed. Aug. 5 @ 11:59 PM**

NEXT TIME...



- Laravel Models & Controllers

- **Due:** Seussology DB II - *TONIGHT*
- **Due:** Hybrid #3 - *TONIGHT*
- **Due:** Seussology - *NEXT WEEK*
- **Due:** Midterm Reflection - *NEXT WEEK*