# Práctica UD-5 Sesión-3

| | |
|---|---|
| Objetivos | • Aprender conceptos de **vistas** y motor de **plantillas Blade** para el desarrollo de aplicaciones webs con Laravel. |
| Instrucciones de entrega | • No hay que entregar nada. |

Abre una línea de comandos y ve a la ruta donde tiene el directorio ***htdocs*** de Xampp (c:\xampp\htdocs) después crea un nuevo proyecto con nombre ***PHPPracticaUD5Sesion3***:

- A través de composer

```
C:\xampp\htdocs>composer create-project laravel/laravel
PHPPracticaUD5Sesion3
```

- O con el comando de Laravel directamente:

```
C:\xampp\htdocs>laravel new PHPPracticaUD5Sesion3
```

- Después entre dentro de la carpeta ***PHPPracticaUD5Sesion3*** recién creada e instale los paquetes ***npm*** que trae Laravel por defecto para evitar el aviso en *NetBeans*:

```
C:\xampp\htdocs\PHPPracticaUD5Sesion3>npm install
```

- Comprobamos que todo ha ido arrancando el servidor embebido:

```
C:\xampp\htdocs\PHPPracticaUD5Sesion3>php artisan serve
```

- Después navegamos a la URL:
  - http://127.0.0.1:8000

- Una vez hecho eso importa el proyecto en NetBeans o en tu IDE o editor favorito.

## Vistas

- Las vistas separan la lógica de su controlador / aplicación de su lógica de presentación y se almacenan en el directorio de ***resources/views***, si utilizamos el motor de plantillas **Blade** tendrá la extensión ***.blade.php*** pero también podemos añadir archivos normales ***php***.

- Veamos un ejemplo, crea un nuevo archivo ***saludo.php*** en el directorio resources/views con el siguiente contenido:

```
<body>
    <h1>¡¡¡Hola!!!</h1>
</body>
```

- Luego edita el fichero **web.php** para añadir la siguiente ruta:

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

- Prueba el resultado navegando a la URL:
    - http://127.0.0.1:8000/saludarRuta

- Crea un nuevo controlador de nombre ***SaludarController*** con un método único y edita el contenido con lo siguiente:

```
return view('saludo');
```

- Añade la ruta para poder llamar al controlador:

```
use App\Http\Controllers\SaludarController;

Route::get('/saludarControlador', SaludarController::class);
```

- Prueba el resultado navegando a la URL:
    - http://127.0.0.1:8000/saludarControlador

### Directorios de vistas anidadas

- Las vistas pueden estar en directorios anidados dentro de la carpeta principal resources/views y accederemos a ellas con el nombre del directorio y un punto (.), por ejemplo cree un nuevo directorio llamado ***admin*** y dentro de él cree una vista ***index.php*** con el siguiente contenido:

```
    <body>
        <h1>Admin home</h1>
    </body>
```

- Añade la siguiente ruta para poder llamarla:

```
Route::get('/admin', function () {
    return view('admin.index');
});
```

- Prueba el resultado navegando a la URL:
  - http://127.0.0.1:8000/admin

## Pasar datos a vistas

- Se puede pasar un array de datos a las vistas para que estos estén disponibles, veamos un ejemplo para ello crea una nueva vista con nombre ***saludocondatos.blade.php*** con el siguiente contenido:

```
    <body>
        <h1>¡¡¡Hola {{$nombre}}!!!</h1>
        <br>
        <h1>¡¡¡Hola <?php echo $nombre ?>!!!</h1>
    </body>
```

- Después añade la siguiente ruta:

```
Route::get('/saludarConDatos', function () {
    return view('saludocondatos', ['nombre' => 'Victoria']);
});
```

- Prueba el resultado navegando a la URL:
  - http://127.0.0.1:8000/saludarConDatos

- Como alternativa a pasar un array completo de datos a la función auxiliar de la vista, se puede usar el método ***with*** para agregar datos individuales, por ejemplo crea una nueva ruta con esta definición:

```
Route::get('/saludarConDatosWith', function () {
    return view('saludocondatos')->with('nombre', 'Victoria Federica');
});
```

- Prueba el resultado navegando a la URL:
  - http://127.0.0.1:8000/saludarConDatosWith

# Motor de plantillas Blade

- Blade es el motor de plantillas simple pero poderoso que se incluye con Laravel. A diferencia de algunos motores de plantillas PHP, Blade no le impide utilizar código PHP directamente en las páginas.

- Las declaraciones ***echo = {{}}*** de Blade se envían automáticamente a través de la función ***htmlspecialchars*** de PHP para evitar ataques XSS.

- Puedes poner cualquier código PHP que desee dentro de una declaración de eco de Blade:
    - `The current UNIX timestamp is {{ time() }}.`

## Directivas

- Blade proporciona accesos directos para estructuras de control PHP comunes, como declaraciones condicionales y bucles. Estos atajos proporcionan una forma muy clara y concisa de trabajar con las estructuras de control de PHP al mismo tiempo que siguen siendo familiares.

- Veamos algunos ejemplos, crea una página ***directivas.blade.php*** con el siguiente contenido:

```
<body>
    @if (count($records) === 1)
    I have one record!
    @elseif (count($records) > 1)
    I have multiple records!
    @else
    I don't have any records!
    @endif

    <br>
    @isset($records)
    // $records is defined and is not null...
    @endisset

    <br>
    @empty($records)
    // $records is "empty"...
    @endempty
</body>
```

- Añade la siguiente ruta:

```
Route::get('/directivasBlade', function () {
    return view('directivas', ['records' => array()]);
});
```

- Prueba el resultado navegando a la URL:
  - http://127.0.0.1:8000/directivasBlade

- Podemos ver el resto de directivas en el manual oficial:
  - https://laravel.com/docs/8.x/blade#blade-directives

## Layouts

- Es conveniente definir un diseño común y luego usarlo en toda nuestra aplicación.

Layout via herencia

- Para crear un layout vamos a utilizar HTML normal y además las directivas ***@section*** y ***@yield***. La directiva *@section*, como su nombre indica, define una sección de contenido, mientras que la directiva *@yield* se usa para mostrar el contenido de una sección determinada.

- Crea un nuevo directorio ***layouts*** dentro de ***views*** y añade un nuevo archivo ***app.blade.php*** con el siguiente contenido:

```
<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>
```

- Ahora vamos a definir una vista secundaria y en ella utilizaremos la directiva ***@extends*** de Blade para especificar qué diseño debe "heredar" la vista secundaria. Las vistas que extienden un diseño Blade pueden inyectar contenido en las secciones del diseño utilizando las directivas @section. Recuerde, como se ve en el ejemplo anterior, el contenido de estas secciones se mostrará en el diseño usando @yield:

- Crea un nuevo archivo ***child.blade.php*** con el siguiente contenido:

```
@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
```

- Ahora añade la ruta para poder probar:

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

- Prueba el resultado navegando a la URL:
    - http://127.0.0.1:8000/layoutporherencia

## Formulario y Bootstrap

- Vamos ahora a crear una página de formulario y le vamos a añadir bootstrap, para ello cree una nueva página ***formulario.blade.php*** con el siguiente contenido:

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Formulario en laravel</title>
```

```
        <meta name="description" content="formulario en laravel">
    </head>
<body>
<div class="container">
<div class="col-sm-8">
<h1>Formulario en laravel</h1>

<form action="/guardar" method="POST">

  <div class="form-group row">
    <label for="inputvehiculo" class="col-sm-2
col-form-label">Vehiculo</label>
    <div class="col-sm-10">
      <input type="text" class="form-control" id="inputvehiculo"
placeholder="Auto o Camioneta">
    </div>
  </div>

  <div class="form-group row">
    <label for="inputmodelo" class="col-sm-2
col-form-label">Modelo</label>
    <div class="col-sm-10">
      <input type="text" class="form-control" id="inputmodelo"
placeholder="Modelo">
    </div>
  </div>

  <div class="form-group row">
    <label for="inputPuertas" class="col-sm-2
col-form-label">Puertas</label>
    <div class="col-sm-10">
      <input type="text" class="form-control" id="inputPuertas"
placeholder="Puertas">
    </div>
  </div>

  <div class="form-group row">
    <label for="inputLuces" class="col-sm-2
col-form-label">Luces</label>
    <div class="col-sm-10">
      <input type="text" class="form-control" id="inputLuces"
placeholder="Luces">
    </div>
  </div>
```

```
  <div class="form-group row">
    <div class="col-sm-10">
      <div class="form-check">
        <input class="form-check-input" name="direccion_asistida"
type="checkbox" id="lbl_da">
        <label class="col-sm-2 form-check-label" for="lbl_da">
          direccion asistida
        </label>
      </div>
    </div>
  </div>

  <div class="form-group row">
    <div class="col-sm-10">
      <div class="form-check">
        <input class="form-check-input" name="abs" type="checkbox"
id="lbl_abs">
        <label class="col-sm-2 form-check-label" for="lbl_abs">
          ABS
        </label>
      </div>
    </div>
  </div>

  <div class="form-group row">
    <div class="col-sm-10">
      <div class="form-check">
        <input class="form-check-input" name="airbags" type="checkbox"
id="lbl_airbags">
        <label class="col-sm-2 form-check-label" for="lbl_airbags">
          Airbags
        </label>
      </div>
    </div>
  </div>

  <div class="form-group row">
    <div class="col-sm-10">
      <button type="submit" class="btn btn-primary">Sign in</button>
    </div>
  </div>
</form>

</div>
</div>
<script src="/js/app.js"></script>
```

```
</body>
</html>
```

- Ahora añade la ruta para poder probar:

```
Route::view('/formulario','formulario');
```

- Prueba el resultado navegando a la URL:
  - http://127.0.0.1:8000/formulario

- Para añadir **bootstrap** tenemos que hacer lo siguiente por línea de comandos:

```
C:\xampp\htdocs\PHPPracticaUD5Sesion3>composer require laravel/ui
C:\xampp\htdocs\PHPPracticaUD5Sesion3>php artisan ui bootstrap
C:\xampp\htdocs\PHPPracticaUD5Sesion3>npm install && npm run dev
```

- Si nos da algún aviso tenemos que volver a invocar a npm install:

```
C:\xampp\htdocs\PHPPracticaUD5Sesion3>npm install && npm run dev
```

- Una vez terminado el proceso edita la página formulario.blade.php añadiendo lo siguiente:

```
<head>
...
<!-- Bootstrap core CSS -->
<link href="/css/app.css" rel="stylesheet">
</head>
<body>
....
<script src="/js/app.js">
</script>
</body>
</html>
```

- Pruebe de nuevo el resultado navegando a la URL:
  - http://127.0.0.1:8000/formulario