La forma de trabajar en Laravel es a través de Blade que es un generador de plantillas web.

1.- Crear la plantilla principal – plantilla.blade.php

Definimos una plantilla principal en la cual se van a basar el resto de vista. Esta plantilla por convenio reside en /resources/views/layouts que vamos a denominar plantilla.blade.php.

Tendremos que crear el subdirectorio layouts (plantillas).

Se adjunta una plantilla que contiene un ejemplo de menú de navegación, denominado nav.blade.php y un ejemplo de un pie de página footer.blade.php. Estas plantillas forman parte de la plantilla principal que tenemos que definir. Creamos un subdirectorio dentro de layouts que denominaremos partials donde ubicaremos ambos archivos.

A la plantilla tenemos que indicarle que utilice estilos y en nuestro caso serán estilos basados en tailwindcss.

Tailwindcss en Laravel utiliza Vite (**Vite** es una herramienta frontend de nueva generación que permite crear un servidor de desarrollo y compilar (hacer el build) del código para producción. También conocida como ViteJS).

Directivas:

- directiva @vite para indicar la ruta del CSS
- directiva @include para que incluya el encabezado y pie de página
- directiva @yield con un nombre asociado, son zonas del documento donde vamos a permitir contenido variable dependiendo de la vista

Un ejemplo de plantilla

resources > views > layouts > plantilla.blade.php > ...

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>@yield('titulo')</title>
    @vite('resources/css/app.css')
</head>
<body>
    @include('layouts.partials.nav')
    <div class="container">
        @yield('contenido')
    </div>
    @include('layouts.partials.footer')
</body>
</html>
```

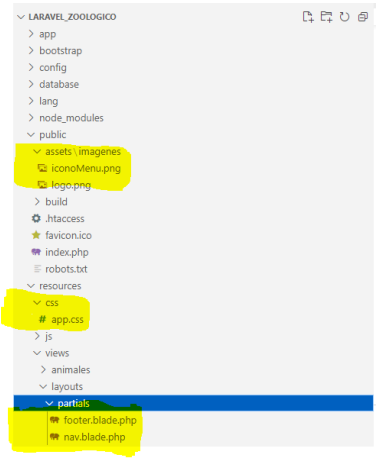
La plantilla de navegación tiene unos estilos que han sido creados en app.css

resources > css > # app.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;
.bverde{
    @apply text-white bg-green-500 border border-green-500 py-2.5 px-5 rounded-md hover:bg-green-600 hover:border-green-600 transition duration-500 ease-in-out;
}
.bblanco{
    @apply text-white border border-white py-2.5 px-5 rounded-md hover:bg-white hover:text-green-800 transition duration-500 ease-in-out;
}
```

También tenemos que subir las imágenes utilizas en la cabecera de la web. Las ubicamos dentro del directorio público (public) en el subdirectorio assets\imagenes:

public\assets\imagenes\iconoMenu.png y public\assets\imagenes\logo.png



2.- Modificar la vista definida en la práctica anterior para que trabaje con la plantilla

La vista se encuentra en el directorio resources/views

resources > views > inicio.blade.php > ...

```
1  @extends('layouts.plantilla')
2  @section('titulo','Zoologico')
3  @section('contenido')
4  <h1 class="text-3xl font-bold underline">Página principal del Zoológico</h1>
5  @endsection
```

Directivas:

- Directiva @extends, indicamos que utiliza la plantilla definida
- Directiva @section, añadimos el contenido de esa sección para cada uno e los @yield de la plantilla

3.- Para organizar mejor las vistas las vamos a agrupar en subcarpetas dentro de la carpeta resources/views siguiendo la siguiente estructura:

Vista	Carpeta	Ruta asociada	Controlador
inicio.blade.php	resources/views/	/	InicioController
index.blade.php	resources/views/animales/	animales	AnimalController
show.blade.php	resources/views/animales/	animales/{animal}	AnimalController
create.blade.php	resources/views/animales/	animales/crear	AnimalController
edit.blade.php	resources/views/animales/	animales/{animal}/editar	AnimalController

Creamos una vista separada para cada ruta.

Por último, vamos a actualizar **los métodos de los controladores**. Acordaros que para referenciar las vistas que están dentro de carpetas la barra / de separación se transforma en un punto y que, además, como segundo parámetro, podemos pasar datos a la vista. A continuación, se incluyen algunos ejemplos:

return view('inicio');

return view('animales.index');

return view('animales.show', ['animal'=>$animal]);

Una vez hechos estos cambios ya podemos probarlo en el navegador, el cual debería mostrar en todos los casos la plantilla base con la barra de navegación principal y los estilos de tailwindcss aplicados. En la sección principal de contenido de momento solo podremos ver los textos que hemos puesto de ejemplo.