

TAREA UT08 - DWES

ANTONIO SANZ PANS
IES MAESTRE DE CALATRAVA

Contenido

Enunciado de la tarea	2
Preparación del entorno	3
Instalación de XAMP y PHP.....	3
Instalación de Composer.....	4
Instalación de NODE.js.....	7
Instalación de GIT	8
Laravel.....	9
Instalación de Laravel	9
Ejecutar Laravel	10
Extensiones útiles para VS	11
Crear rutas en Laravel	12
Controladores	13
Vistas.....	16
Plantillas Blade	17
Recorrido final	18
Tarea 07 Extra	19
Conectar con la BBDD	19
Listar alumnos	19
Crear registros	21
Editar registros	23

ENUNCIADO DE LA TAREA

Como se ha indicado en la introducción, en este último tema solo se persigue que se adquiriera un conocimiento elemental de la estructura y funcionamiento de Laravel. Por esto en la tarea correspondiente a este tema se va a pedir lo siguiente:

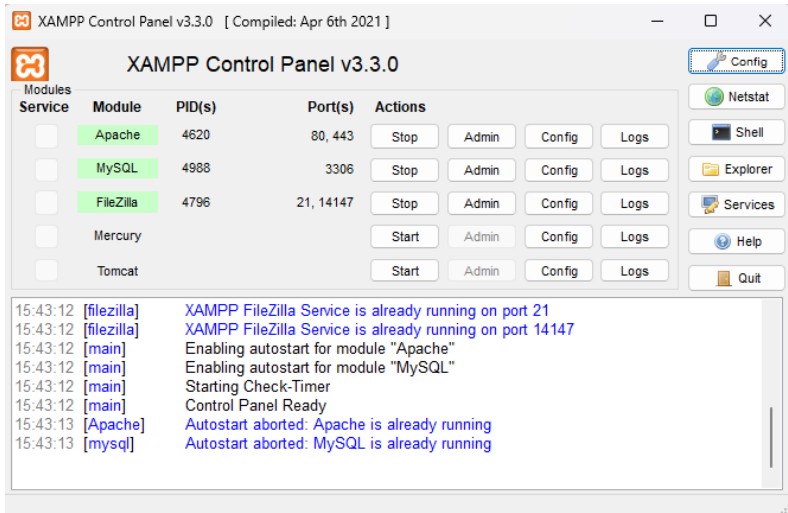
- Realizar una instalación de Laravel (versión 10 u 11), creando un proyecto llamado 'tarea7' mostrando y explicando en un documento *.pdf las capturas de pantalla en las que se detalle el proceso.
- Modificar la vista wellcome que viene por defecto con un mensaje personalizado.
- Conectar el proyecto con la base de datos de la tarea anterior e indicar como se ha realizado(0,5p)
- Crear una vista llamada /listar que muestre en una tabla html todos los alumnos de la tabla alumnos.sql.(1,5p)
- Crear una vista llamada /alta que permita insertar alumnos en la tabla alumnos.sql.(1,5p)
- Crear una vista llamada /actualizar que permita actualizar datos de un alumno en la tabla alumnos.sql.(1,5p)

Para los últimos apartados será necesario entregar además del documento pdf el código comprimido del proyecto Laravel creado.

PREPARACIÓN DEL ENTORNO

Instalación de XAMP y PHP

Como durante el curso esto ha sido un requisito, adjunto capturas de que están instalados, pero no de su instalación.

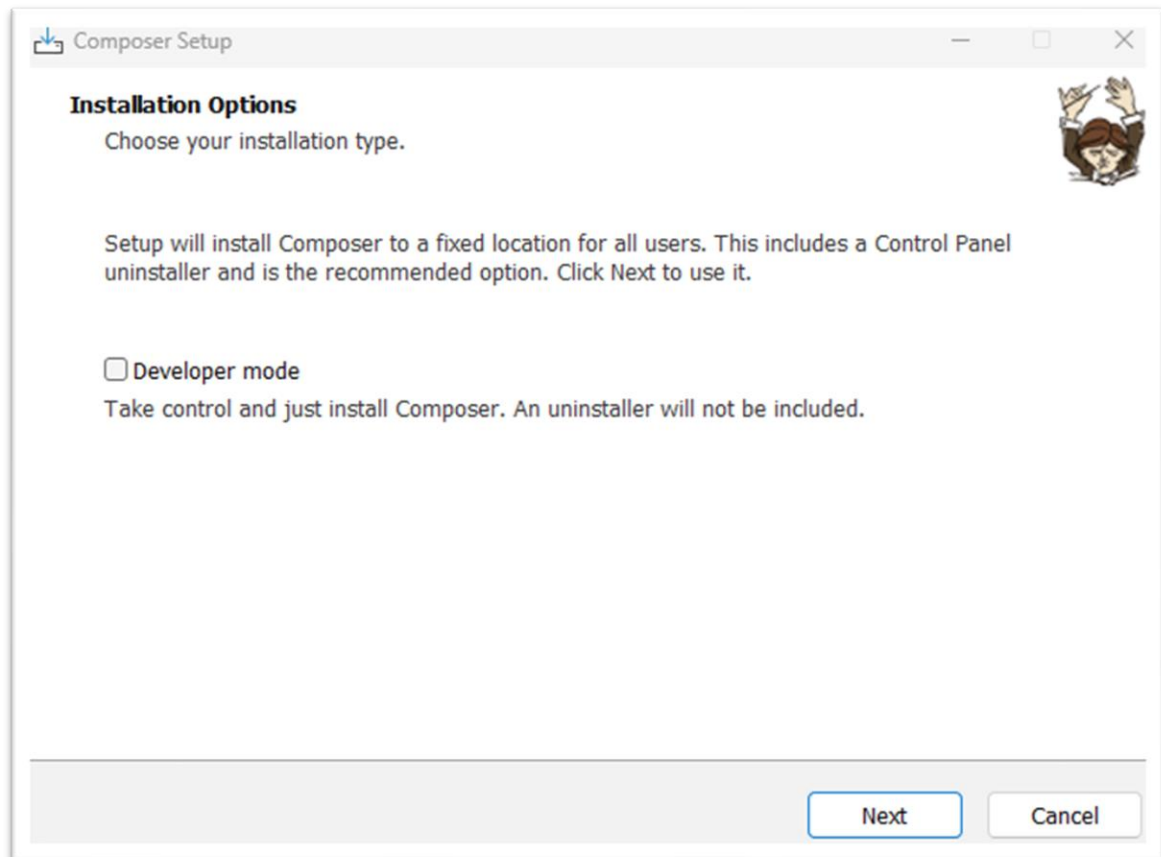


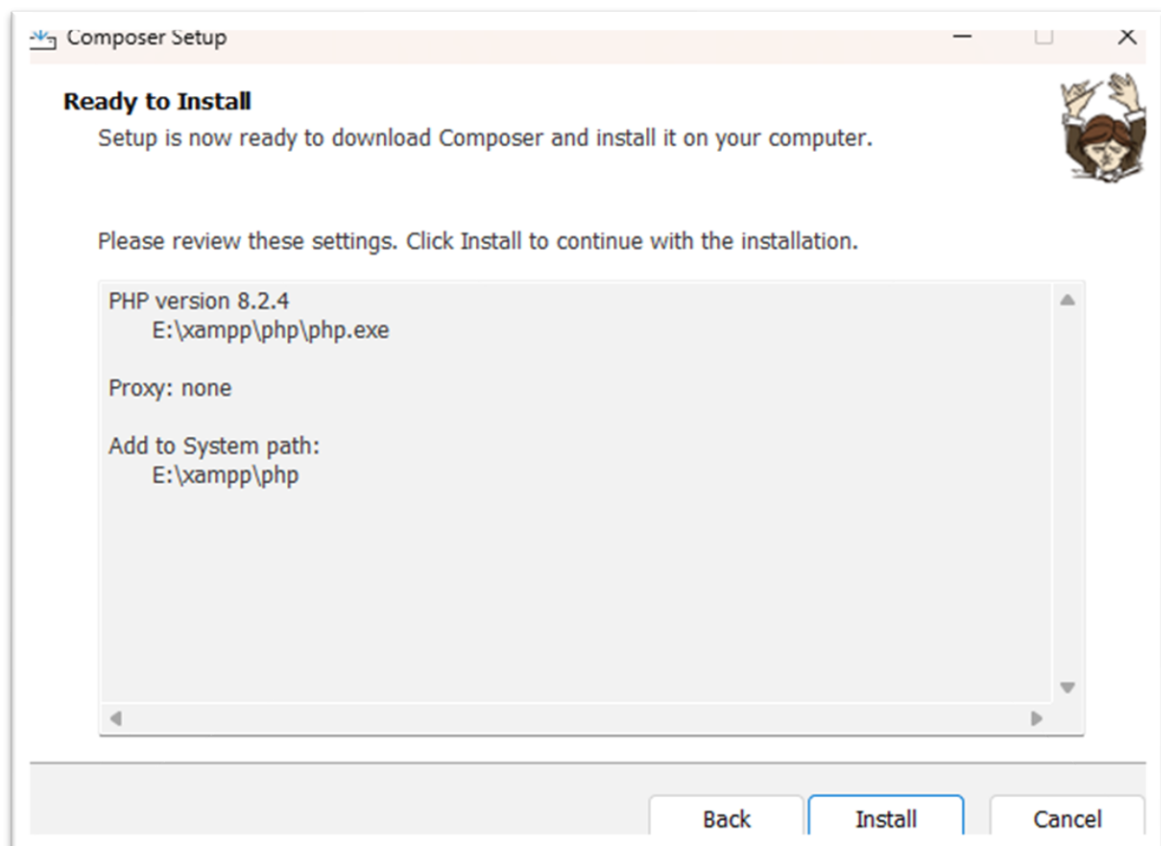
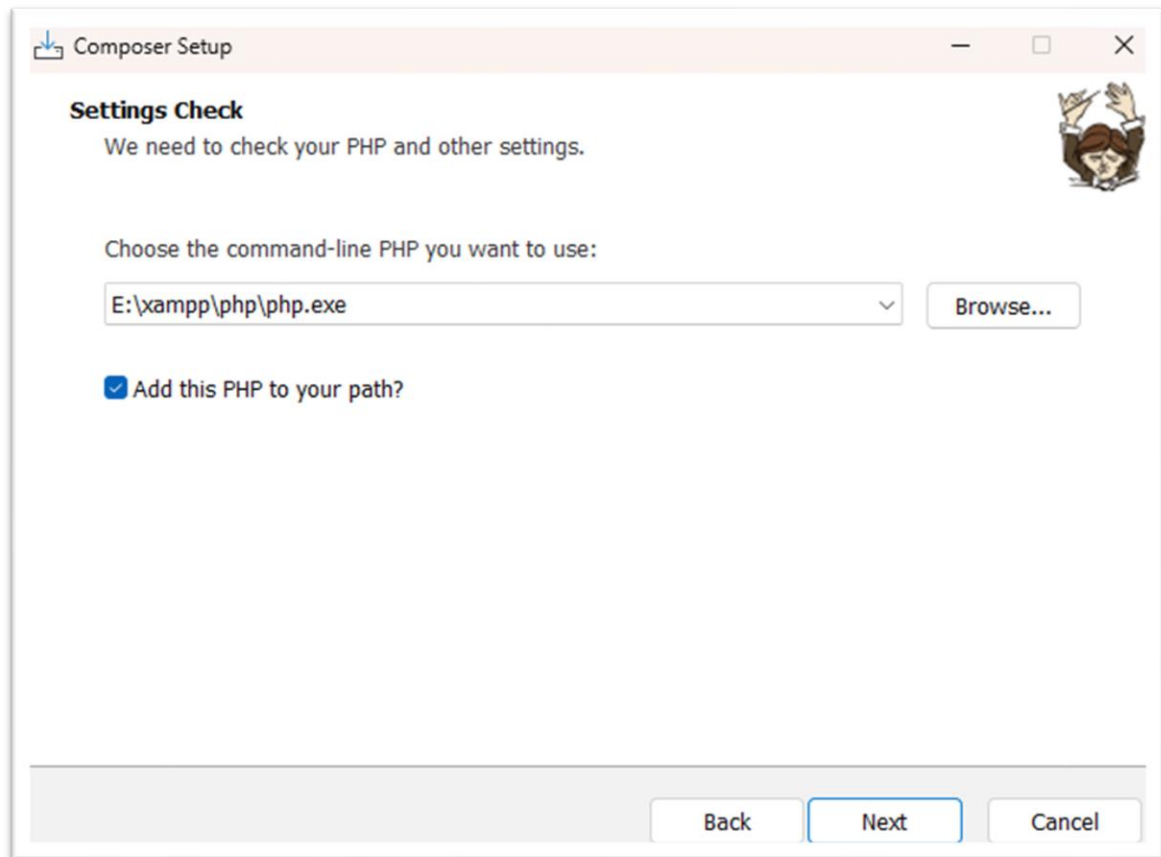
PHP Version 8.2.4

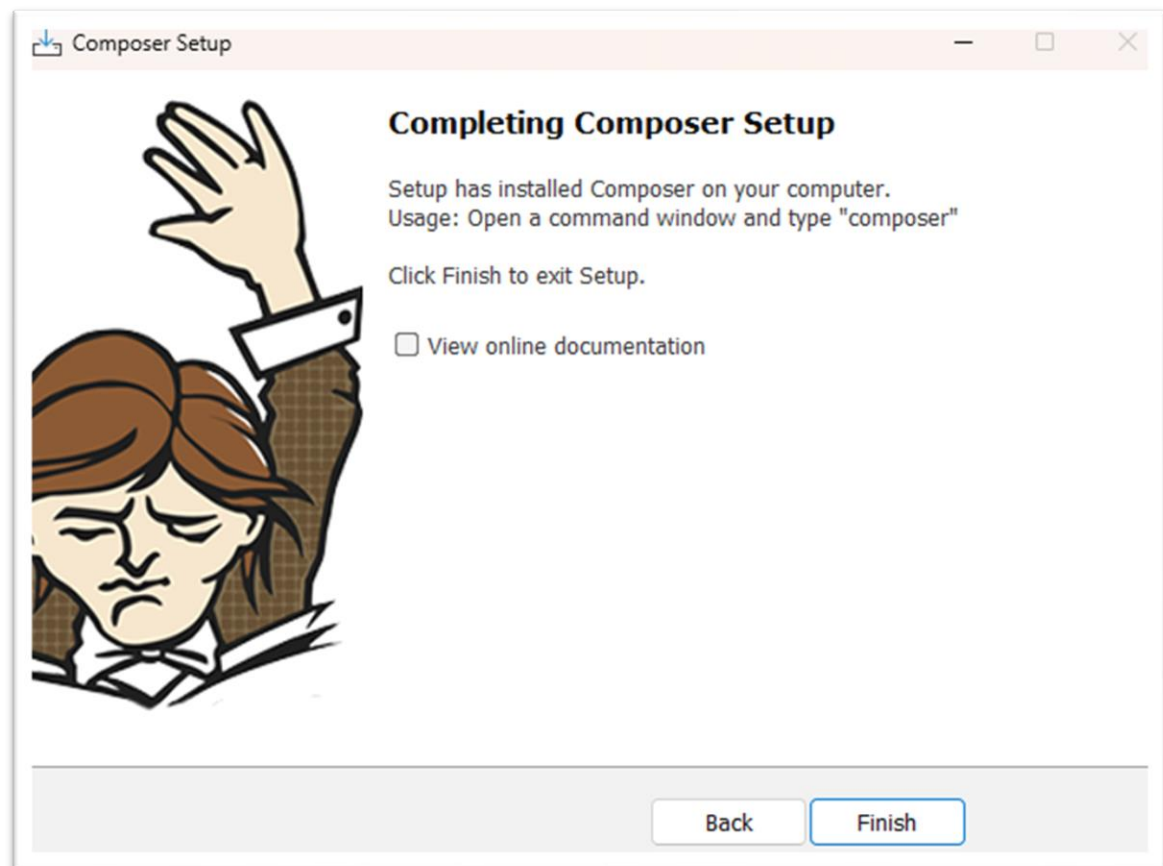
System	Windows NT PC-ANTO 10.0 build 22631 (Windows 11) AMD64
Build Date	Mar 14 2023 17:50:26
Build System	Microsoft Windows Server 2019 Datacenter [10.0.17763]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cscrip /nologo /e:jscrip configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=.\\..\\..\\instantclient\\sdk,shared" "--with-oci8-19=.\\..\\..\\instantclient\\sdk,shared" "--enable-object-out-dir=.\\obj" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	no value
Loaded Configuration File	E:\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20220829
PHP Extension	20220829
Zend Extension	420220829
Zend Extension Build	API420220829,TS,VS16
PHP Extension Build	API20220829,TS,VS16
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	php, file, glob, data, http, ftp, compress.zlib, compress.bzip2, https, ftps, phar
Registered Stream Socket Transports	tcp, udp, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	convert.iconv.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, zlib.*, bzip2.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v4.2.4. Copyright (c) Zend Technologies
with Xdebug v3.3.1, Copyright (c) 2002-2023, by Derick Rethans

Instalación de Composer

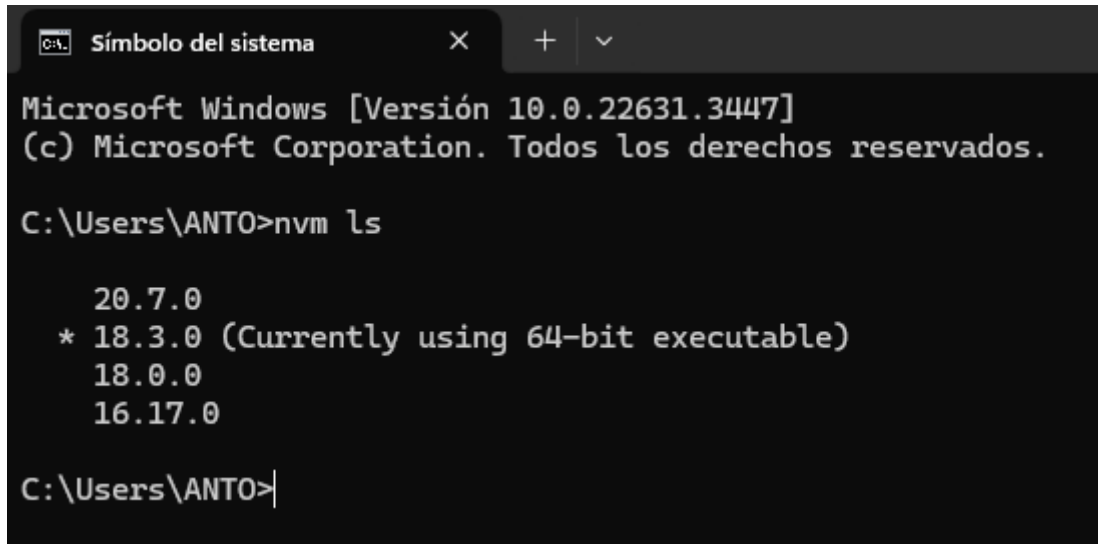






Instalación de NODE.js

Yo ya tenía instalado tanto Node como un gestor de versiones (NVM). Para esto utilizaré la ultima versión de node.



```
C:\> Símbolo del sistema X + v

Microsoft Windows [Versión 10.0.22631.3447]
(c) Microsoft Corporation. Todos los derechos reservados.

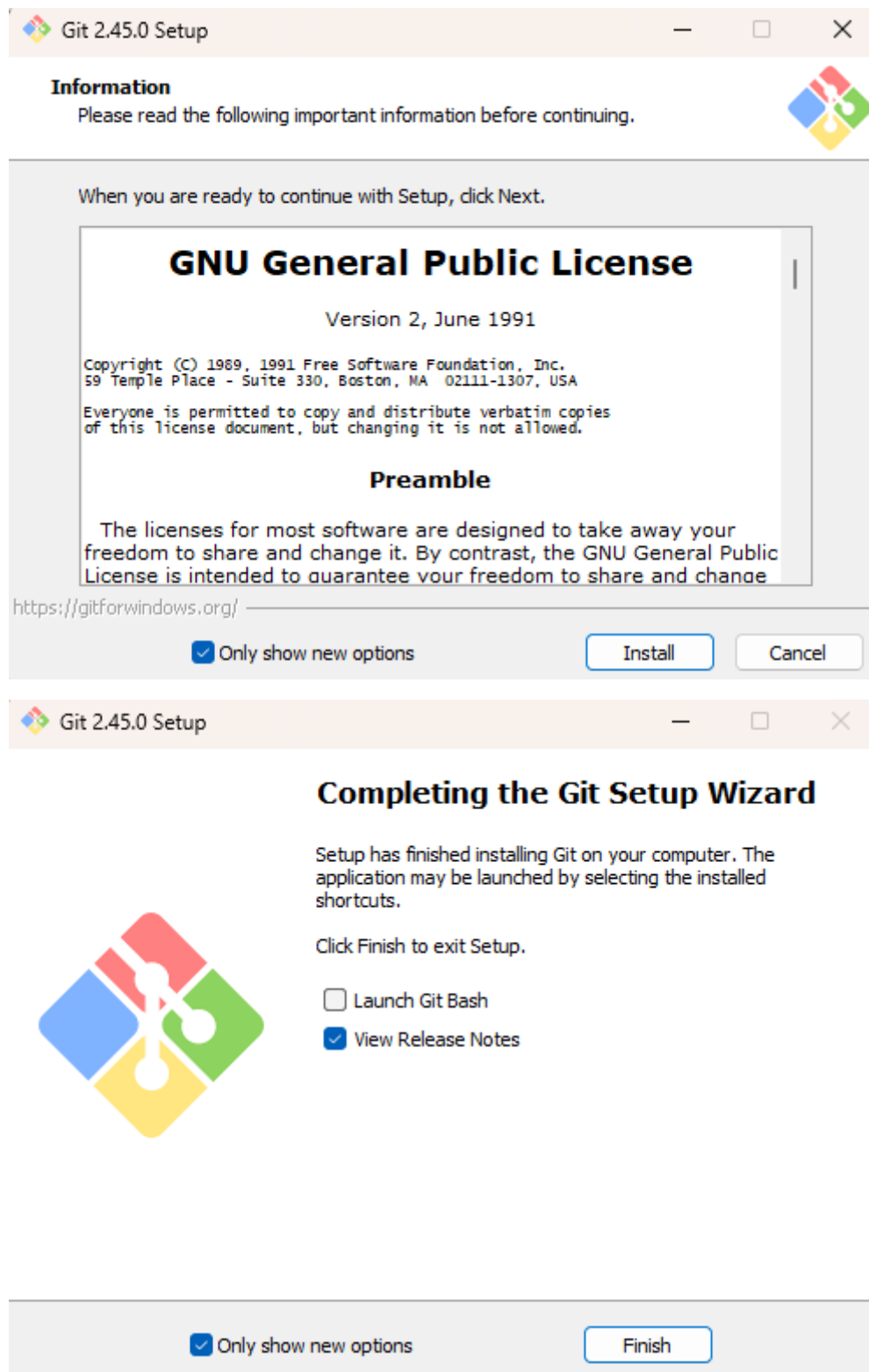
C:\Users\ANTO>nvm ls

 20.7.0
* 18.3.0 (Currently using 64-bit executable)
 18.0.0
 16.17.0

C:\Users\ANTO>|
```


Instalación de GIT

Aunque yo uso GitHub, instalare GIT para seguir el curso y que se instale GitShell.



LARAVEL

Instalación de Laravel

Para crear el proyecto con Laravel debemos movernos a la carpeta de xampp/htdocs y utilizar el comando:

```
composer create-project laravel/laravel example-app
```

```
MINGW64:/e/xampp/htdocs
ANTO@PC-ANTO MINGW64 /e/xampp/htdocs
$ composer create-project laravel/laravel dwes-tarea7|
```

```
INFO Discovering packages.
laravel/sail ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE
spatie/laravel-ignition ..... DONE

85 packages you are using are looking for funding.
Use the 'composer fund' command to find out more!
> @php artisan vendor:publish --tag=laravel-assets --ansi --force

INFO No publishable resources for tag [laravel-assets].

No security vulnerability advisories found.
> @php artisan key:generate --ansi

INFO Application key set successfully.

> @php -r "file_exists('database/database.sqlite') || touch('database/database.sqlite');"
> @php artisan migrate --graceful --ansi

INFO Preparing database.

Creating migration table ..... 16.82ms DONE

INFO Running migrations.

0001_01_01_000000_create_users_table ..... 61.02ms DONE
0001_01_01_000001_create_cache_table ..... 20.00ms DONE
0001_01_01_000002_create_jobs_table ..... 49.98ms DONE

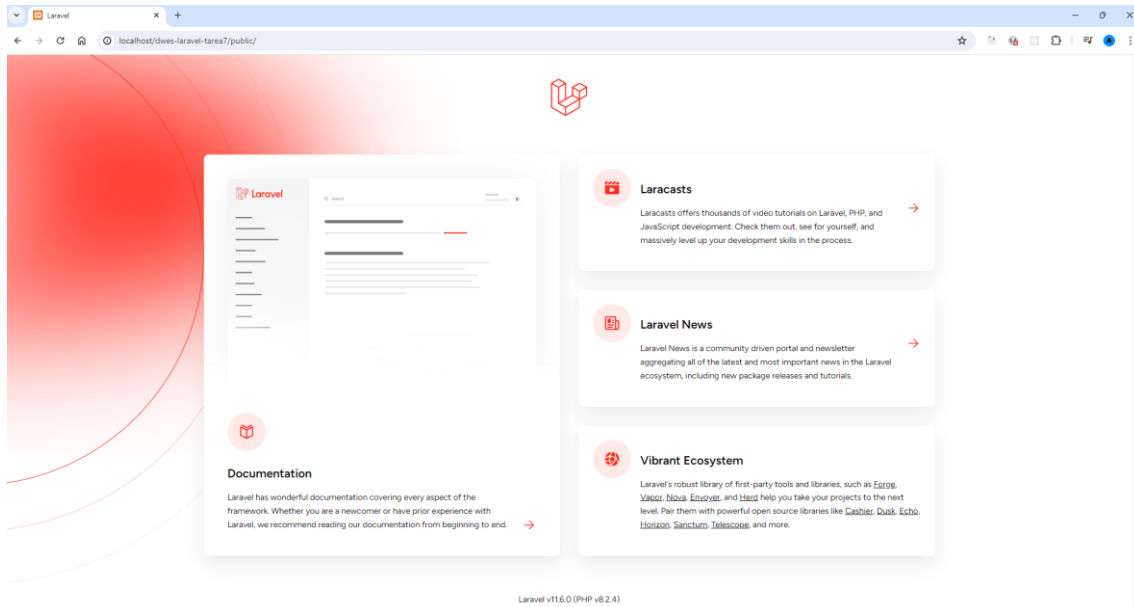
ANTO@PC-ANTO MINGW64 /e/xampp/htdocs
$ |
```

Ejecutar Laravel


Lanzar apache mediante XAMPP



Acceder a nuestro Proyecto desde el navegador desde
[localhost/\[NombreDelProyectoLaravel\]/public/](localhost/[NombreDelProyectoLaravel]/public/)




Extensiones útiles para VS




Laravel Blade formatter


v0.24.2

Shuhei Hayashibara [shufo.dev](#) | 1.357.641 | ★★★★★ (26) |  Patrocinador

Laravel Blade formatter for VSCode

[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.




Laravel Blade Snippets


v1.36.0

Winnie Lin | 3.358.930 | ★★★★★ (36)

Laravel blade snippets and syntax highlight support

[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.




Laravel goto view


v1.3.11

codingyu | 1.642.752 | ★★★★★ (21)

Quick jump to view

[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.




Laravel Snippets


v1.18.0

Winnie Lin | 2.221.216 | ★★★★★ (11)

Laravel snippets for Visual Studio Code (Support Laravel 5 and above)


[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.




PHP Intelephense


v1.10.4

Ben Mewburn [intelephense.com](#) | 11.679.243 | ★★★★★ (374) |  Patrocinador

PHP code intelligence for Visual Studio Code

[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.




Tailwind CSS IntelliSense


v0.10.5

Tailwind Labs [tailwindcss.com](#) | 6.072.162 | ★★★★★ (95)

Intelligent Tailwind CSS tooling for VS Code

[Deshabilitar](#) [Desinstalar](#) [Cambiar a la versión preliminar](#) 

Esta extensión está habilitada globalmente.




Alpine.js IntelliSense

v1.2.0

Adrian Wilczyński | 139.660 | ★★★★★ (5)

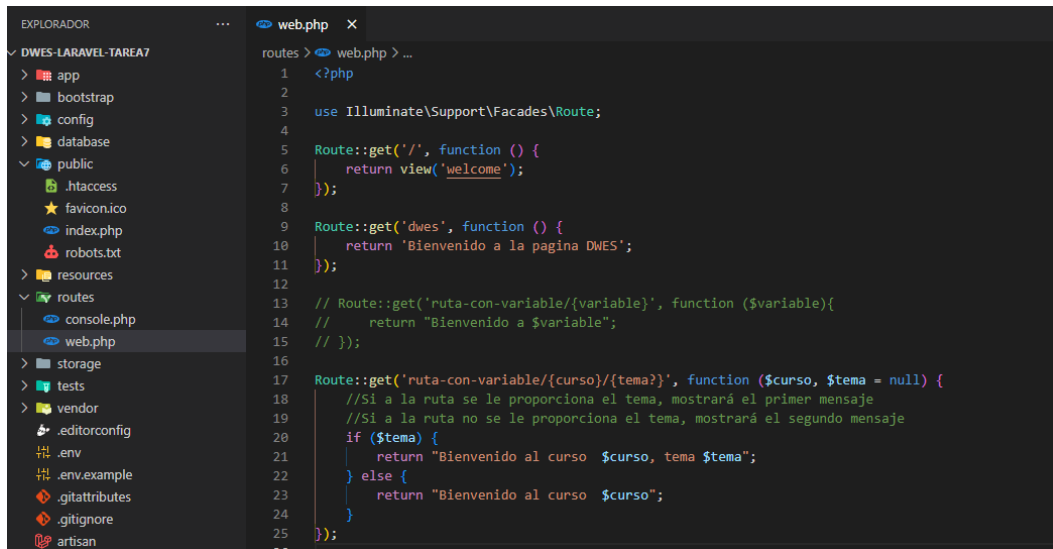
Simple IntelliSense & Snippets for Alpine.js framework.

[Deshabilitar](#) [Desinstalar](#) 

Esta extensión está habilitada globalmente.

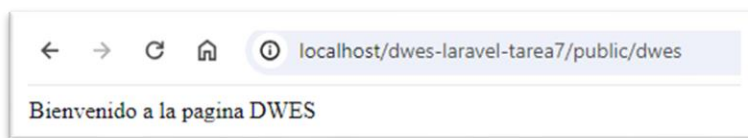
Crear rutas en Laravel

Para crear rutas en laravel ha que establecerlas dentro de la carpeta `routes/web.php`



```
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4
5 Route::get('/', function () {
6     return view('welcome');
7 });
8
9 Route::get('dwes', function () {
10     return 'Bienvenido a la pagina DWES';
11 });
12
13 // Route::get('ruta-con-variable/{variable}', function ($variable){
14 //     return "Bienvenido a $variable";
15 // });
16
17 Route::get('ruta-con-variable/{curso}/{tema?}', function ($curso, $tema = null) {
18     //Si a la ruta se le proporciona el tema, mostrará el primer mensaje
19     //Si a la ruta no se le proporciona el tema, mostrará el segundo mensaje
20     if ($tema) {
21         return "Bienvenido al curso $curso, tema $tema";
22     } else {
23         return "Bienvenido al curso $curso";
24     }
25 });
26
```

Ruta estática



Ruta con variable



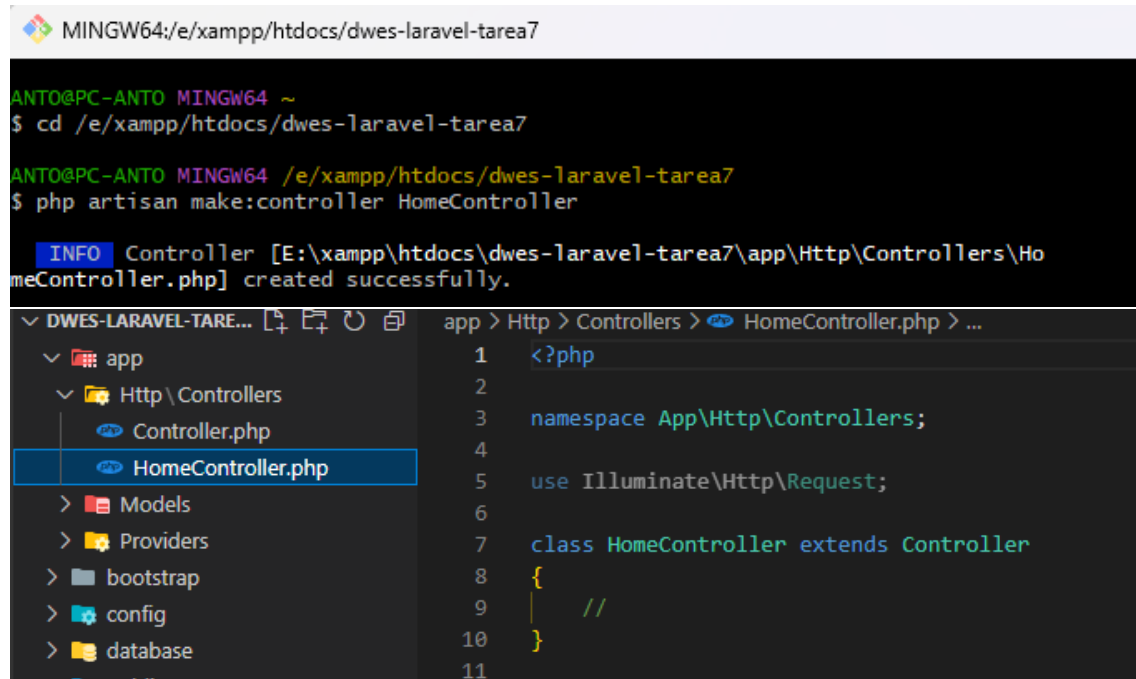
Ruta con variable opcional



Controladores

Los controladores deben de ir en app/Http/Controllers

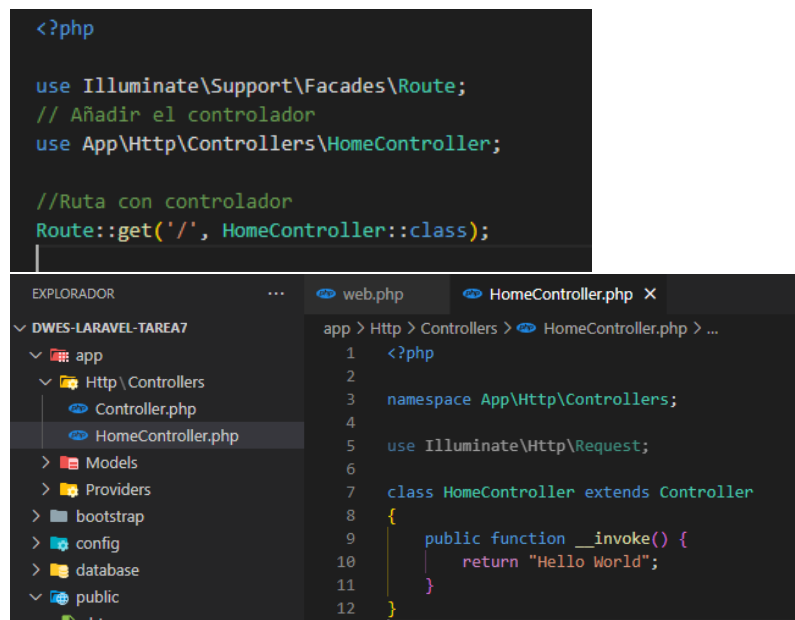
La mejor forma para crear un controlador es desde la terminal ejecutando `php artisan make:controller [NombreControlador]`



The screenshot shows a terminal window at the top with the command `php artisan make:controller HomeController` being executed. Below the terminal, the VS Code editor shows the file explorer on the left with `app > Http > Controllers > HomeController.php` selected. The main editor area shows the content of `HomeController.php`:

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class HomeController extends Controller
8 {
9     //
10 }
11
```

Añadir un controlador para la ruta home



The screenshot shows two parts of the VS Code editor. The top part shows a code snippet for adding a route:

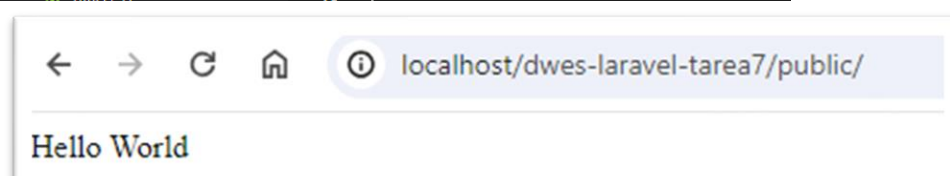
```
<?php

use Illuminate\Support\Facades\Route;
// Añadir el controlador
use App\Http\Controllers\HomeController;

//Ruta con controlador
Route::get('/', HomeController::class);
```

The bottom part shows the file explorer with `app > Http > Controllers > HomeController.php` selected. The main editor area shows the content of `HomeController.php` with the `__invoke()` method added:

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class HomeController extends Controller
8 {
9     public function __invoke() {
10         return "Hello World";
11     }
12 }
```

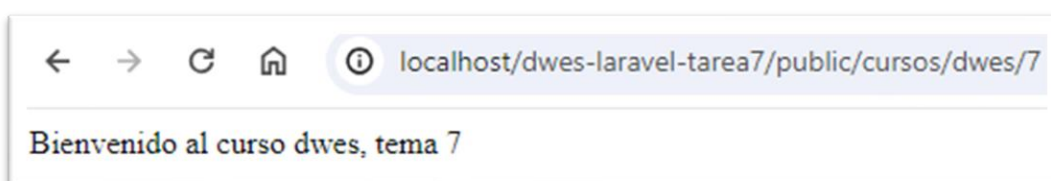
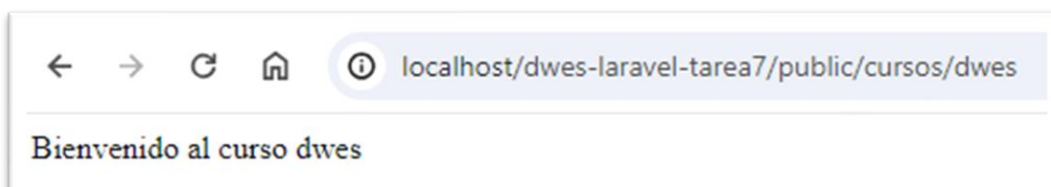
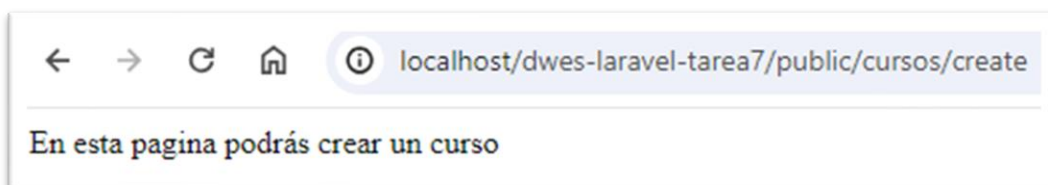
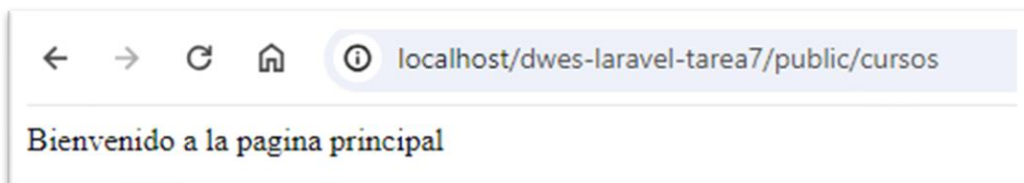
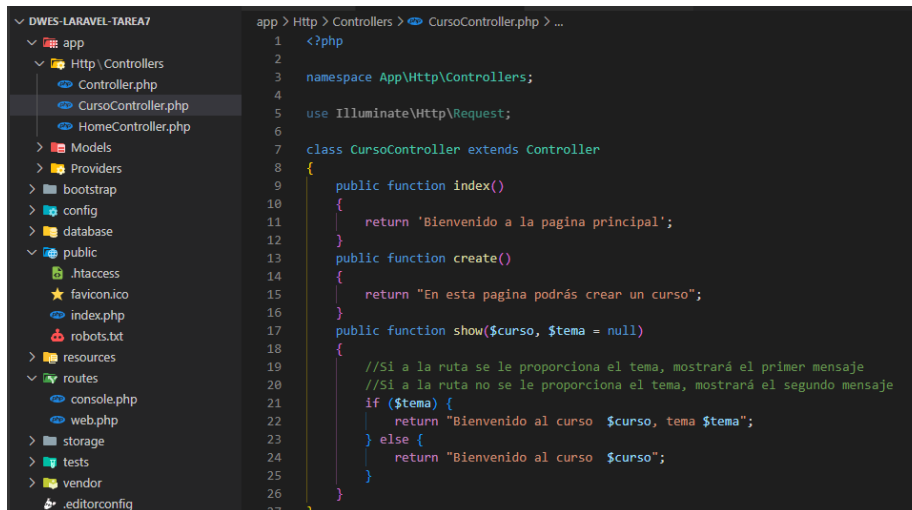


Añadir controlador para el resto de rutas

```
<?php

use Illuminate\Support\Facades\Route;
// Añadir el controlador
use App\Http\Controllers\HomeController;
use App\Http\Controllers\CursoController;

//Ruta con controlador
Route::get('/', HomeController::class);
Route::get('cursos', [CursoController::class, 'index']);
Route::get('cursos/create', [CursoController::class, 'create']);
Route::get('cursos/{curso}/{tema?}', [CursoController::class, 'show']);
```



Rutas agrupadas

```
<?php

use Illuminate\Support\Facades\Route;
// Añadir el controlador
use App\Http\Controllers\HomeController;
use App\Http\Controllers\CursoController;

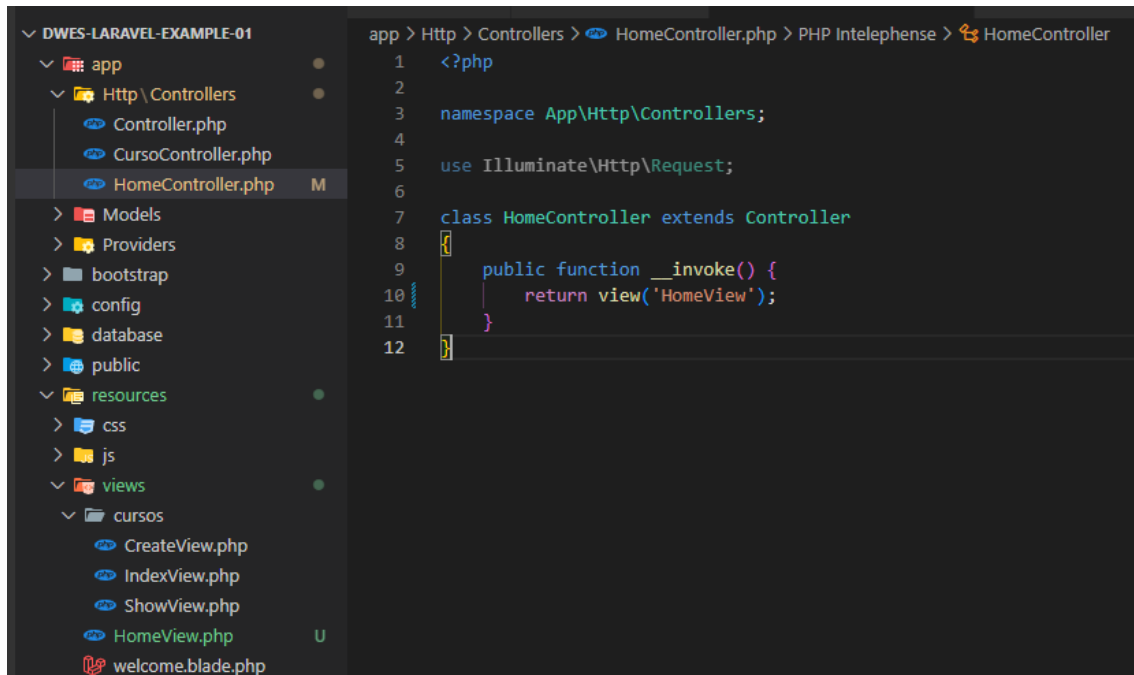
//Ruta con controlador
Route::get('/', HomeController::class);

//Grupo de rutas
Route::controller(CursoController::class)->group(function(){
    Route::get('cursos', 'index');
    Route::get('cursos/create', 'create');
    Route::get('cursos/{curso}/{tema?}', 'show');
});
```


Vistas

Las vistas se deben crear en la carpeta **resources/views**. En mi caso las organizaré en carpetas dentro de views.

Modificando el controlador **HomeController** que está enlazado a la raíz del proyecto y añadiendo la vista **HomeView** puedo modificar la pagina inicial de mi proyecto.



The screenshot shows an IDE with a file explorer on the left and a code editor on the right. The file explorer displays the project structure for 'DWES-LARAVEL-EXAMPLE-01', with the 'resources' folder expanded to show 'views' and 'cursos' subfolders. The 'HomeController.php' file is selected. The code editor shows the following PHP code:

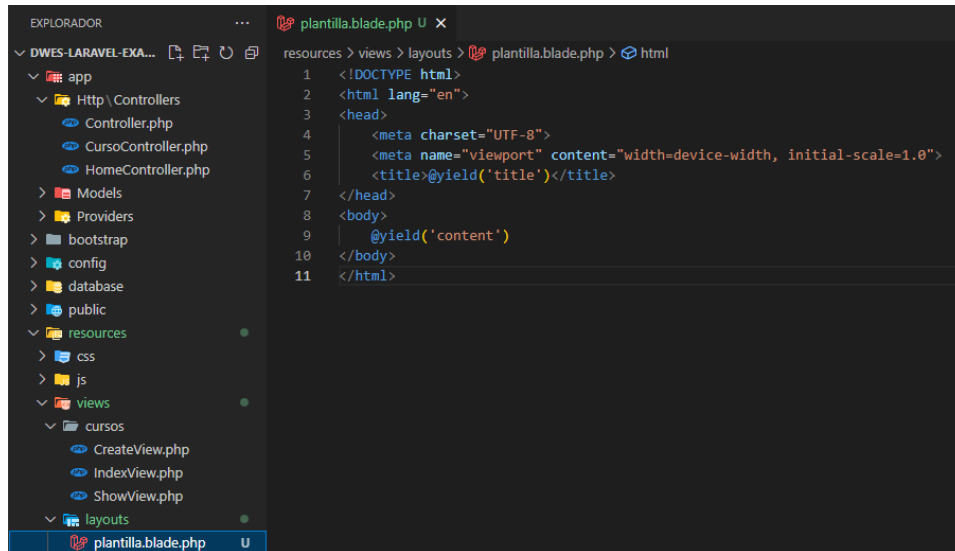
```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class HomeController extends Controller
8 {
9     public function __invoke() {
10         return view('HomeView');
11     }
12 }
```



Plantillas Blade

Para crear plantillas debemos crear la carpeta `resources/views/layouts` y dentro crear archivos con la extensión `.blade.php`

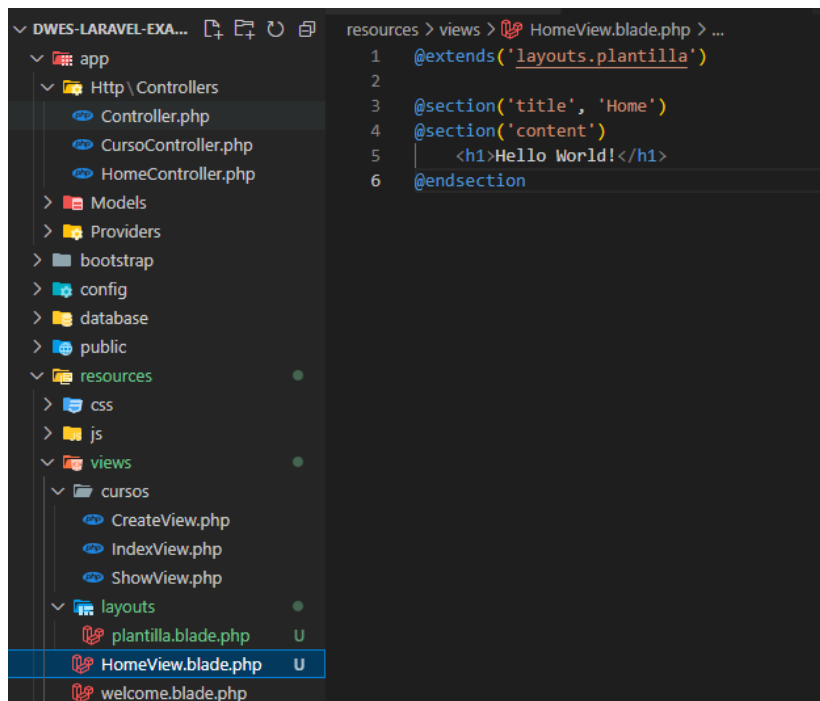
En la plantilla se añade el código HTML que vamos a reutilizar y declaramos los campos variables con `@yield('nombreDelCampo')`



The screenshot shows the VS Code interface with the Explorer on the left and the Editor on the right. The Explorer shows the project structure with the `resources/views/layouts` folder selected. The Editor shows the content of `plantilla.blade.php`, which is a basic HTML template with a head section containing meta tags and a body section with a `@yield('content')` placeholder.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>@yield('title')</title>
7 </head>
8 <body>
9   @yield('content')
10 </body>
11 </html>
```

Los archivos PHP debemos sustituir su extensión `.php` a `.blade.php` y su contenido por `@section('Nombre del campo de la plantilla', 'Contenido que se muestra')`

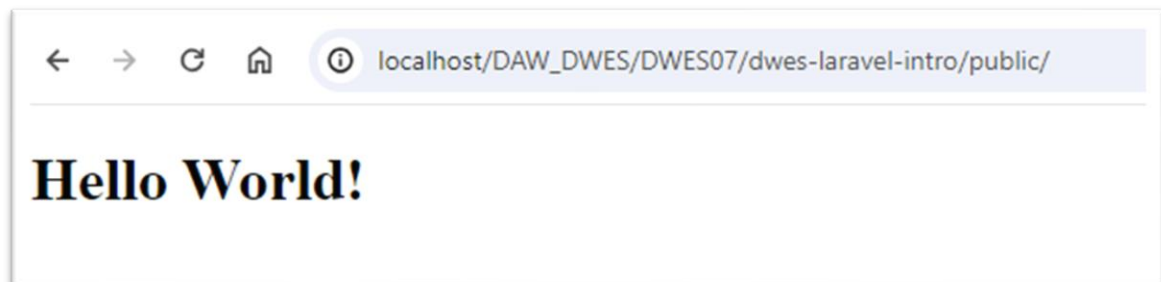


The screenshot shows the VS Code interface with the Explorer on the left and the Editor on the right. The Explorer shows the project structure with the `resources/views/layouts` folder selected. The Editor shows the content of `HomeView.blade.php`, which extends the `layouts.plantilla` template and defines a `@section('title', 'Home')` and a `@section('content')` with the text `<h1>Hello World!</h1>`.

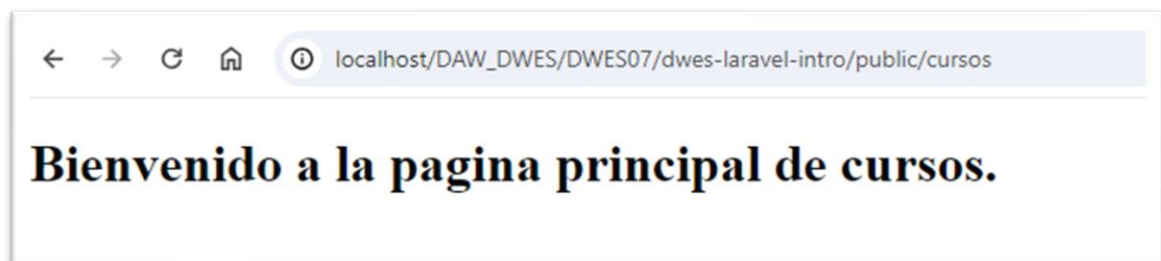
```
1 @extends('layouts.plantilla')
2
3 @section('title', 'Home')
4 @section('content')
5   <h1>Hello World!</h1>
6 @endsection
```

Recorrido final

Página inicial a la que accedemos (Página welcome modificada) (/)



Página principal creada para cursos (/cursos)



Página creada para un curso concreto (/cursos/DWES) que cambia según reciba el parámetro del curso



TAREA 07 EXTRA

Conectar con la BBDD

Hay que modificar el archivo `config/database.php`

```
database.php M x .env
config > database.php
5 return [
15 | is explicitly specified when you execute a query / statement.
16 |
17 */
18
19 // 'default' => env('DB_CONNECTION', 'sqlite'),
20 'default' => env('DB_CONNECTION', 'mysql'),
21
```

Añadir credenciales a `.env`

```
database.php M .env x
.env
17 LOG_CHANNEL=stack
18 LOG_STACK=single
19 LOG_DEPRECATIONS_CHANNEL=null
20 LOG_LEVEL=debug
21
22 DB_CONNECTION=mysql
23 DB_HOST=127.0.0.1
24 DB_PORT=3306
25 DB_DATABASE=daw_dwes_07
26 DB_USERNAME=root
27 DB_PASSWORD=
```

Listar alumnos

Primero crear el modelo para obtener los datos de la tabla Alumno con `php artisan make:model Alumno`

```
app > Models > Alumno.php > ...
1 <?php
2
3 namespace App\Models;
4
5 // use Illuminate\Database\Eloquent\Factories\HasFactory;
6 use Illuminate\Database\Eloquent\Model;
7
8 class Alumno extends Model
9 {
10     // use HasFactory;
11     protected $table = 'alumnos'; // Nombre de la tabla en la base de datos
12
13 }
```

Despues del modelo, generar el controlador para alumno y enlazarlo con la vista correspondiente

```
app > Http > Controllers > AlumnoController.php > ...
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\Alumno; // Importa el modelo Alumno
7
8 class AlumnoController extends Controller
9 {
10     public function listarAlumnos()
11     {
12         $alumnos = Alumno::all(); // Obtener todos los registros de la tabla 'alumnos'
13         return view('ListarView', ['alumnos' => $alumnos]);
14     }
15 }
16
17
```

```
resources > views > ListarView.blade.php > html
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Lista de Alumnos</title>
5 </head>
6 <body>
7     <h1>Lista de Alumnos</h1>
8     <ul>
9         @foreach ($alumnos as $alumno)
10             <li>{{ $alumno->Nombre }}</li>
11         @endforeach
12     </ul>
13 </body>
14 </html>
```

Por último añadirlo en el archivo web.php

```
routes > web.php > ...
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4 // Añadir el controlador
5 use App\Http\Controllers\HomeController;
6 use App\Http\Controllers\CursoController;
7 use App\Http\Controllers\AlumnoController;
8
9 //Ruta con controlador
10 Route::get('/', HomeController::class);
11 Route::get('/listar', [AlumnoController::class, 'listarAlumnos']);
12
13 //Grupo de rutas
14 Route::controller(CursoController::class)->group(function(){
15     Route::get('cursos', 'index');
16     Route::get('cursos/create', 'create');
17     Route::get('cursos/{curso}/{tema?}', 'show');
18 });
19
```

Este es el resultado

Lista de Alumnos	
DNI	Nombre
11111111A	juan
22222222B	luis
33333333C	tomas
44444444D	jose
55555555E	antonio
66666666F	oscar
77777777G	carlos

Crear registros

Creo las rutas necesarias para crear alumnos y enlazarlas con sus vistas

```
//Crear Alumno
Route::get('alumnos/crear', [AlumnoController::class, 'mostrarFormularioCrear']) -> name('alumnos.crear'); //Formulario de crear
Route::post('alumnos', [AlumnoController::class, 'crearAlumno']) -> name('alumnos.guardar'); //Ruta para crear Alumno
```

La vista para crear alumno

```
resources > views > alumnos > crearView.blade.php > ...
1  @extends('layouts.plantilla')
2
3  @section('title', 'Crear Alumno')
4
5  @section('content')
6      <div class="container">
7          <h1>Crear Nuevo Alumno</h1>
8          <form action="{{route('alumnos.guardar')}}" method="POST">
9              {{-- TOKEN --}} @csrf
10             <div class="mb-3">
11                 <label for="dni" class="form-label">DNI:</label>
12                 <input type="text" class="form-control" id="dni" name="dni" required maxlength=9>
13             </div>
14             <div class="mb-3">
15                 <label for="nombre" class="form-label">Nombre:</label>
16                 <input type="text" class="form-control" id="nombre" name="nombre" required maxlength=50>
17             </div>
18             <div class="mb-3">
19                 <label for="apellido1" class="form-label">Apellido 1:</label>
20                 <input type="text" class="form-control" id="apellido1" name="apellido1" required maxlength=50>
21             </div>
22             <div class="mb-3">
23                 <label for="apellido2" class="form-label">Apellido 2:</label>
24                 <input type="text" class="form-control" id="apellido2" name="apellido2" required maxlength=50>
25             </div>
26             <button type="submit" class="btn btn-primary">Guardar</button>
27         </form>
28     </div>
29 @endsection
```

El método del controlador que mandara los datos a la BBDD y redirigirá al listado.

```
class AlumnoController extends Controller
{
    public function crearAlumno(Request $request)
    {
        // return $request;

        //Crear el objeto alumno
        $alumno = new Alumno();
        $alumno->DNI = $request->dni;
        $alumno->Nombre = $request->nombre;
        $alumno->Apellido1 = $request->apellido1;
        $alumno->Apellido2 = $request->apellido2;

        //Guardar en la base de datos
        $alumno->save();

        // // Redirigir a una página de éxito o a otra vista
        return redirect('/alumnos/listar')->with('success', 'Alumno creado correctamente');
    }
}
```

El resultado:

Pantalla del listado con el nuevo botón para crear alumno

Lista de Alumnos

DNI	Nombre	
11111111	juan	editar
123123123	ANTONIO	editar
123456789	ANTONIO FERNANDO	editar
22222222	luis	editar
33333333	tomas	editar
44444444	jose	editar
55555555	antonio	editar
66666666	oscar	editar
77777777	carlos	editar

[crear alumno](#)

El formulario con los datos ya introducidos

Crear Nuevo Alumno

DNI:

Nombre:

Apellido 1:

Apellido 2:

[Guardar](#)

Cuando pulsamos guardar nos redirige al listado con el nuevo registro

Lista de Alumnos

DNI	Nombre	
11111111	juan	editar
123123123	ANTONIO	editar
123456789	ANTONIO FERNANDO	editar
123654789	JUAN	editar
22222222	luis	editar
33333333	tomas	editar
44444444	jose	editar
55555555	antonio	editar
66666666	oscar	editar
77777777	carlos	editar

[crear alumno](#)

Editar registros

Crear los routes para editar un alumno

```
//Editar Alumno
Route::get('alumnos/{dni}/editar', [AlumnoController::class, 'mostrarFormularioEditar']) -> name('alumnos.editar');
Route::put('alumnos/{alumno}', [AlumnoController::class, 'actualizarAlumno']) -> name('alumnos.actualizar');
);
```

Añadir a la vista del listado el botón de editar

```
<tbody>
    @foreach ($alumnos as $alumno)
        <tr>
            <td>{{ $alumno->DNI }}</td>
            <td>{{ $alumno->Nombre }}</td>
            <td><a href="{{ route('alumnos.editar', ['dni' => $alumno->DNI]) }}">editar</a></td>
        </tr>
    @endforeach
</tbody>
```

Añadir un view para la pantalla de editar, que será prácticamente igual que el formulario de crear

```
resources > views > alumnos > @@ editarView.blade.php > div.container > form
1  @extends('layouts.plantilla')
2
3  @section('title', 'Editar Alumno')
4
5  @section('content')
6      <p>{{ $alumno }}</p>
7
8      <div class="container">
9          <h1>Editar Alumno</h1>
10         <form action="{{ route('alumnos.actualizar', $alumno) }}" method="post">
11             @csrf
12             @method('put')
13
14             <div class="mb-3">
15                 <label for="dni" class="form-label">DNI:</label>
16                 <input type="text" class="form-control" id="dni" name="dni" required maxlength=9 value="{{ $alumno->DNI }}">
17             </div>
18             <div class="mb-3">
19                 <label for="nombre" class="form-label">Nombre:</label>
20                 <input type="text" class="form-control" id="nombre" name="nombre" required maxlength=50 value="{{ $alumno->Nombre }}">
21             </div>
22             <div class="mb-3">
23                 <label for="apellido1" class="form-label">Apellido 1:</label>
24                 <input type="text" class="form-control" id="apellido1" name="apellido1" required maxlength=50 value="{{ $alumno->Apellido1 }}">
25             </div>
26             <div class="mb-3">
27                 <label for="apellido2" class="form-label">Apellido 2:</label>
28                 <input type="text" class="form-control" id="apellido2" name="apellido2" required maxlength=50 value="{{ $alumno->Apellido2 }}">
29             </div>
30             <button type="submit" class="btn btn-primary">Editar</button>
31         </form>
32     </div>
33 @endsection
```

Crear los métodos en el controlador de alumnos para editar

```
public function mostrarFormularioEditar($dni)
{
    $alumno = Alumno::find($dni);
    return view('alumnos.editarView', compact('alumno'));
}

public function actualizarAlumno(Alumno $alumno, Request $request)
{
    $alumno -> Dni = $request -> dni;
    $alumno -> Nombre = $request -> nombre;
    $alumno -> Apellido1 = $request -> apellido1;
    $alumno -> Apellido2 = $request -> apellido2;
    $alumno->save();
    return redirect('/alumnos/listar')->with('success', 'Alumno actualizado correctamente');
}
```


Este es el resultado, para empezar la pantalla del listado que ya tenía, pero se agrega el botón de editar.

Lista de Alumnos		
DNI	Nombre	
11111111	juan	editar
123123123	ANTONIO	editar
123456789	ANTO	editar
22222222	luis	editar
33333333	tomas	editar
44444444	jose	editar
55555555	antonio	editar
66666666	oscar	editar
77777777	carlos	editar
crear alumno		

Cuando pulsamos el botón de editar se abre el formulario, ha quedado un poco feo, pero si me da tiempo lo editaré. He dejado el objeto sin modificar arriba en un <p> para que se vea el cambio

```
{ "DNI":123456789,"Nombre":"ANTO","Apellido1":"SANZ","Apellido2":"PANS" }
```

Editar Alumno

DNI:

Nombre:

Apellido 1:

Apellido 2:

A continuación el campo editado, como se puede ver en el objeto sin editar, el nombre en ambos sitios es distinto.

```
{ "DNI":123456789,"Nombre":"ANTO","Apellido1":"SANZ","Apellido2":"PANS" }
```

Editar Alumno

DNI:

Nombre:

Apellido 1:

Apellido 2:

Una vez pulsamos el botón editar, editara el registro y nos devolverá al listado actualizado.

Lista de Alumnos		
DNI	Nombre	
11111111	juan	editar
123123123	ANTONIO	editar
123456789	ANTONIO FERNANDO	editar
22222222	luis	editar
33333333	tomas	editar
44444444	jose	editar
55555555	antonio	editar
66666666	oscar	editar
77777777	carlos	editar
crear alumno		

NOTA: Me he dado cuenta que si edito los registros preexistentes no funciona bien. No sé exactamente por qué pero hay que tenerlo en cuenta.