

BIENVENIDO A LA GUIA CURSO LARAVEL IMPARTIDO EN SOFTURA

Esta guía es una breve introducción a Laravel, en este archivo encontraras una guía de como realizar un CRUD conectado a Base de Datos utilizando Laravel.

PROLOGO

No necesitas ser experto para manejar un determinado lenguaje de programación, incluso personas con el mas alto grado académico “doctorado” no lo aprenden todo en su trayectoria académica, o aplicandolo como docentes, lo principal es enseñar y aprender de ello, pero te preguntaras ¿Cómo lo enseñare? Si apenas estoy aprendiendo. Pues esa es la clave, nunca aprenderás si no compartes el conocimiento.

No se le puede llamar una persona sabia por su inteligencia sino porque comparte su conocimiento.



Ahora que te quieres involucrar a lo laboral, no desaproveches esta oportunidad, no lo hagas solo por cumplir, hazlo por aprender, no importa cuanto te equivoques, cuantas veces preguntes, cuantas veces investigues, cuantas noches no duermas. Hazlo por ti. ¡Tú puedes!

INDICE

Contenido

PROLOGO	1
Objetivo.....	3
Introducción	3
Herramientas de trabajo	3
Configuración de Instalación	4
Creación Proyecto Laravel	5
Configuración Virtual Hosts	8
Creación de Login	12
Creación de Migraciones.....	12
CREACION CRUD.....	13
Creación de vistas.....	16
Estructura proyecto.....	19

Objetivo

Comprender la estructura, funcionamiento y los componentes mas importantes de Laravel creando un CRUD (Create , Read, Update, Delete).

Introducción




Lo que necesitas saber...

1. Este no es solo un proyecto por realizar, sino que del extraigas información relevante y principalmente soluciones dudas a través de esta guía.
2. Nuestros proyecto se van a guardar en una carpeta determinada (generalmente C:\wamp\www en Wamp y C:\xampp\htdocs en el caso de Xamp).
3. Todo sistema operativo presenta un archivo hosts, para guardar la correspondencia entre dominios de Internet y direcciones IP.
4. En Windows, el archivo hosts se ubica en C:\Windows\System32\drivers\etc\hosts.

Herramientas de trabajo

Necesitaras instalar:

1. Repositorio Git https://github.com/Marcos9108/curso_laravel_0822.git
2. Composer <https://getcomposer.org/download/>
3. Wamp <https://www.filehorse.com/es/descargar-wampserver-64/>

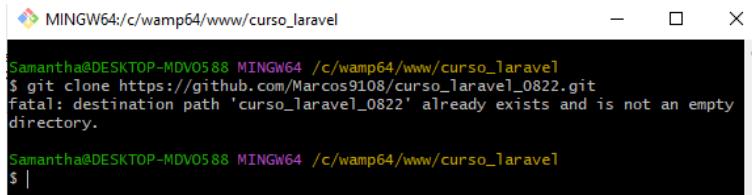
	Apache	2.4.41	Version Apache 2.4.41
	PHP	7.2.25	Version Php 7.2.
	MySQL	5.7.28	Version MySQL 5

4. IDE o editor de código de preferenica , como por ejemplo;
Visual Studio Code <https://code.visualstudio.com/download>

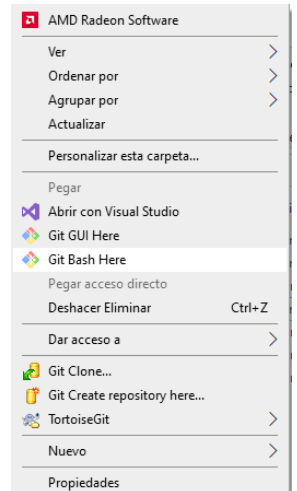
Notepad++ <https://notepad-plus-plus.org/downloads/>
IntelliJ <https://www.jetbrains.com/es-es/idea/download/>

Configuración de Instalación

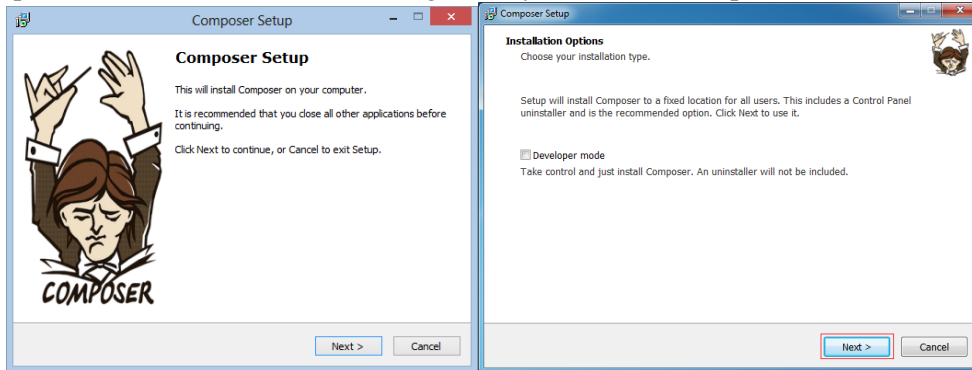
1. Clonación Repositorio Git; Para obtener la información de código deberás clonar el repositorio. Das clic derecho en tu explorador de archivos , seleccionas la opción de Git Bash Here, una vez abierto ejecutar el siguiente comando; git clone y la ruta del repositorio de git



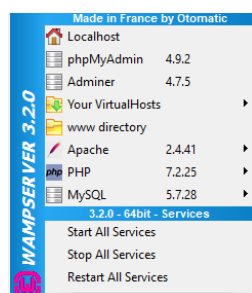
```
MINGW64:/c:/wamp64/www/curso_laravel
Samantha@DESKTOP-MDV0588 MINGW64 /c:/wamp64/www/curso_laravel
$ git clone https://github.com/Marcos9108/curso_laravel_0822.git
fatal: destination path 'curso_laravel_0822' already exists and is not an empty
directory.
Samantha@DESKTOP-MDV0588 MINGW64 /c:/wamp64/www/curso_laravel
$ |
```



2. Descargar WAMP, seleccionar las siguientes versiones;
3. Descargar composer; Ejecuta el asistente de instalación de Composer. Cuando te solicite que actives el modo desarrollador, ignóralo y continúa con el proceso de instalación.



4. Ejecutar composer; una vez descargado composer , dentro del Símbolo de Sistema o bien CMD, se escribira el comando composer, este mandara la versión del mismo y comprobara que se haya instalado correctamente.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 10.0.19042.1089]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\Samantha>composer

Composer version 2.4.1 2022-08-20 11:44:58

Usage:
  command [options] [arguments]

Options:
  -h, --help                Display help for the given command. When no command is given display help for the list command
  -q, --quiet               Do not output any message
  -V, --version             Display this application version
                             Force (or disable --no-ansi) ANSI output
  -n, --no-ansi             Do not ask any interactive question
                             Display timing and memory usage information
                             Whether to disable plugins.
  -d, --working-dir=WORKING-DIR If specified, use the given directory as working directory.
                             Prevent use of the cache
  -vv|vvv, --verbose       Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug

Available commands:
  about          Shows a short information about Composer
  archive        Creates an archive of this composer package
  audit          Checks for security vulnerability advisories for installed packages
  [home]         [Home] Opens the package's repository URL or homepage in your browser
  bump          Increases the lower limit of your composer.json requirements to the currently installed versions
  check-platform-reqs Check that platform requirements are satisfied
  clear-cache    [clear-cache] clears composer's internal package cache
  completion    Dump the shell completion script
  config        Sets config options
  create-project Creates new project from a package into given directory
  depends       [why] Shows which packages cause the given package to be installed
  diagnose      Diagnoses the system to identify common errors
  dump-autoload [dump-autoload] Dumps the autoloader
  exec          Executes a vendored binary/script
  find           Discover how to help find the maintenance of your dependencies
```

Creación Proyecto Laravel

Instalación Laravel; existen dos maneras para crear la carpeta con el framework de laravel

1. La primera opción es utilizando composer , ejecutaremos el siguiente comando;

Composer create-project --prefer-dist laravel/laravel nuevo-proyecto "5.5.*"

```
Simbolo del sistema
C:\wamp64\www>composer create-project --prefer-dist laravel/laravel curso_laravel
```

Asimismo el 5.5 indica la versión de laravel a ocupar.

1.2 Si utilizaste composer, también tendrás que renombrar el archivo creado en tu explorador de archivos , asignado como .env.example y renombrarlo a .env

Este equipo > Disco local (C:) > wamp64 > www > curso_laravel > curso_laravel_0822 >

	Nombre	Fecha de modificación	Tipo	Tamaño
ido	.git	06/09/2022 08:45 a. m.	Carpeta de archivos	
is	app	06/09/2022 08:45 a. m.	Carpeta de archivos	
ntos	bootstrap	06/09/2022 08:45 a. m.	Carpeta de archivos	
Inv. Plantillas	config	06/09/2022 08:45 a. m.	Carpeta de archivos	
loud Files	database	06/09/2022 08:45 a. m.	Carpeta de archivos	
	public	06/09/2022 08:45 a. m.	Carpeta de archivos	
	resources	06/09/2022 08:45 a. m.	Carpeta de archivos	
	routes	06/09/2022 08:45 a. m.	Carpeta de archivos	
	storage	06/09/2022 08:45 a. m.	Carpeta de archivos	
	tests	06/09/2022 08:45 a. m.	Carpeta de archivos	
is	vendor	06/09/2022 09:01 a. m.	Carpeta de archivos	
ntos	artisan	06/09/2022 08:45 a. m.	Archivo	2 KB
	composer.json	06/09/2022 08:59 a. m.	Archivo de origen ...	2 KB
	package.json	06/09/2022 08:45 a. m.	Archivo de origen ...	2 KB
s	readme.md	06/09/2022 08:45 a. m.	Archivo de origen ...	4 KB
3D	server.php	06/09/2022 08:45 a. m.	Archivo de origen ...	1 KB
	.env	06/09/2022 09:12 a. m.	Archivo ENV	1 KB
	webpack.mix.js	06/09/2022 08:45 a. m.	Archivo JavaScript	1 KB
al (C:)	composer.lock	06/09/2022 08:45 a. m.	Archivo LOCK	198 KB
	.gitattributes	06/09/2022 08:45 a. m.	Documento de te...	1 KB
	.gitignore	06/09/2022 08:45 a. m.	Documento de te...	1 KB
	phpunit.xml	06/09/2022 08:45 a. m.	Documento XML	2 KB

1.3 Asimismo tendrás que generar una clave nueva para el proyecto, ahora bien para hacer esto, dirígete a un CMD, y coloca el siguiente comando;

`php artisan key: generate`

```
> @php artisan key:generate
Application key [base64:fLPU2wtp2+0DSAup8FdngM8SqCY2sM2MZ8nh+JGAvg=] set successfully.
```

Esta llave es propia de la aplicación y se ocupa para implementar un método de seguridad, en donde el back y front trabajarán en conjunto a través de un token que reconoce las peticiones de la aplicación.

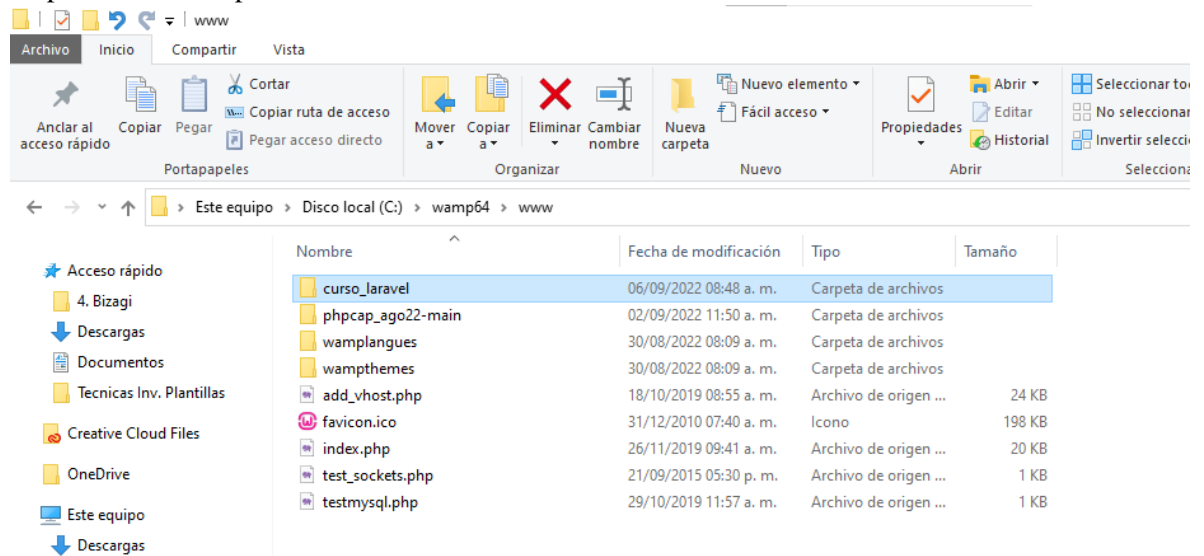
2. La segunda opción es realizando la instalación de Laravel, se ejecuta con el siguiente comando ;

`composer global require "laravel/installer".`

`Laravel new nuevo-proyecto.`

Recuerda que el nombre de tu proyecto lo decides tú, esta es solo una guía.

Estos comandos generaran una carpeta en tu explorador de archivos que tendrá la instalación básica del framework, con estos comandos ya no es necesario volver a recrear la carpeta desde tu explorador de archivos.



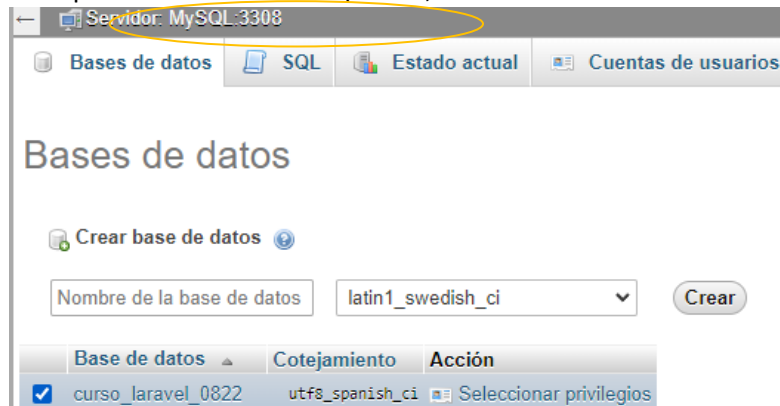
Creación Base de Datos

1. Acceder como root a la base de datos

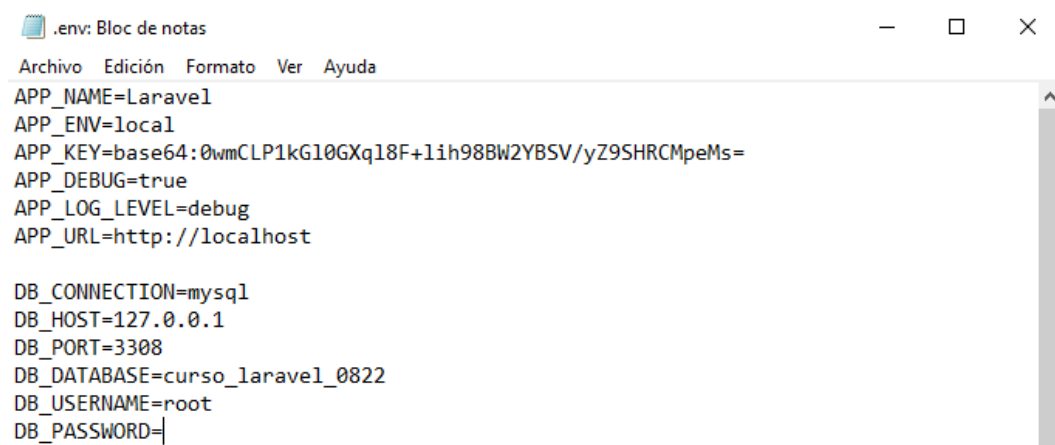


2. Crear una BD con las siguientes características ; utf8_spanish_ci.

Es importante considerar el puerto, en mi caso 3308.



3. Conectar Laravel con nuestra BD; para ello requerimos acceder a nuestro archivo .env y cambiar esta linea de codigo por un valor TRUE y agregamos los valores de nuestra base de datos



Configuración Virtual Hosts

1. Concepto Hosts Virtual ;

Primero que nada , antes de crearlo debes saber que es , y para que se utiliza, pues bien...

Virtual host es la herramienta que Apache pone a nuestra disposición para poder publicar más de una página web en un mismo servidor Apache, permitiéndonos mantener configuraciones diferentes para cada host virtual.

Los VirtualHosts son útiles en distintos escenarios.

En este caso, los usaremos para configurar nuestro entorno de desarrollo local.

2. Configurar VirtualHosts con Apache

Como en todo hay múltiples opciones, si ya tienes mas experiencia, puedes saltarte esta parte , pero recomiendo que lo lleves a cabo;

2.1 Primera opción “Usar Apache con la configuración manual de VirtualHosts”.

Apache tiene un archivo de configuración. Debes abrir este archivo.

En el caso de Wamp se ubica en: C:\wamp\bin\apache\apache2.4.41\conf\httpd.conf

Y en el caso de Xampp: C:\xampp\apache\conf\httpd.conf

Accedemos al archivo C:\wamp64\bin\apache\apache2.4.41\conf\httpd.conf

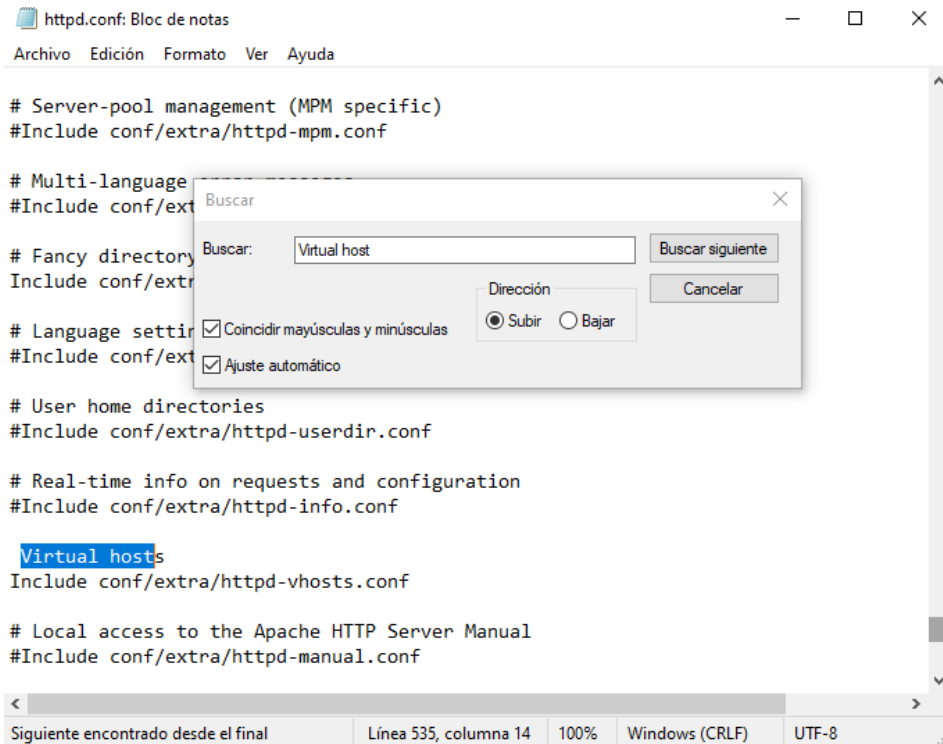
Este equipo > Disco local (C:) > wamp64 > bin > apache > apache2.4.41 > conf				
	Nombre	Fecha de modificación	Tipo	Tamaño
do	extra	30/08/2022 08:11 a. m.	Carpeta de archivos	
	original	30/08/2022 08:21 a. m.	Carpeta de archivos	
	charset.conv	09/08/2019 03:50 p. m.	Archivo CONV	2 KB
ios	httpd.conf	11/09/2022 11:41 a. m.	Archivo CONF	22 KB
iv. Plantillas	magic	09/08/2019 03:50 p. m.	Archivo	14 KB
ud Files	mime.types	09/08/2019 03:50 p. m.	Archivo TYPES	62 KB
	openssl.cnf	28/05/2019 02:12 p. m.	Archivo CNF	11 KB

Editamos las siguientes lineas de codigo desde el bloc de notas;

```
# Virtual hosts
```

```
Include conf/extra/httpd-vhosts.conf
```

Eso significa que la declaración de Virtual hosts la podemos realizar en conf/extra/httpd-vhosts.conf (dentro de la carpeta de apache donde ya estamos ubicados).



Si tenemos un mismo proyecto Laravel que atiende distintas direcciones, también podemos configurar ello en un mismo VirtualHost del siguiente modo:

```
<VirtualHost *:8080>
    ServerName localhost.com
    ServerAlias series.localhost.com memes.localhost.com foro.localhost.com
    DocumentRoot "C:/wamp/www/pym/public"
</VirtualHost>
```

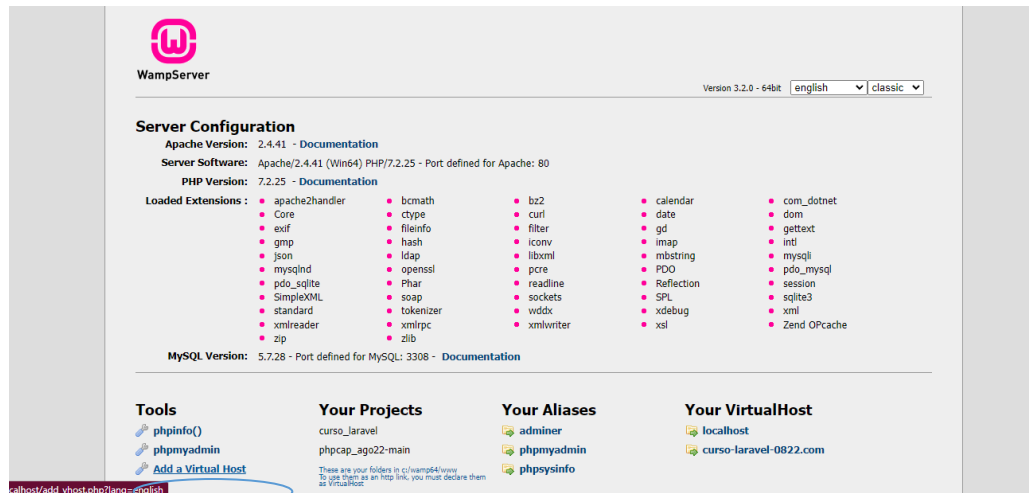
Las direcciones especificadas en ServerAlias (separadas por un espacio) se van a atender por el mismo DocumentRoot, al igual que la dirección indicada en el ServerName.

Si tu proyecto Laravel no tiene rutas para distintos subdominios, siempre mostrará lo mismo.

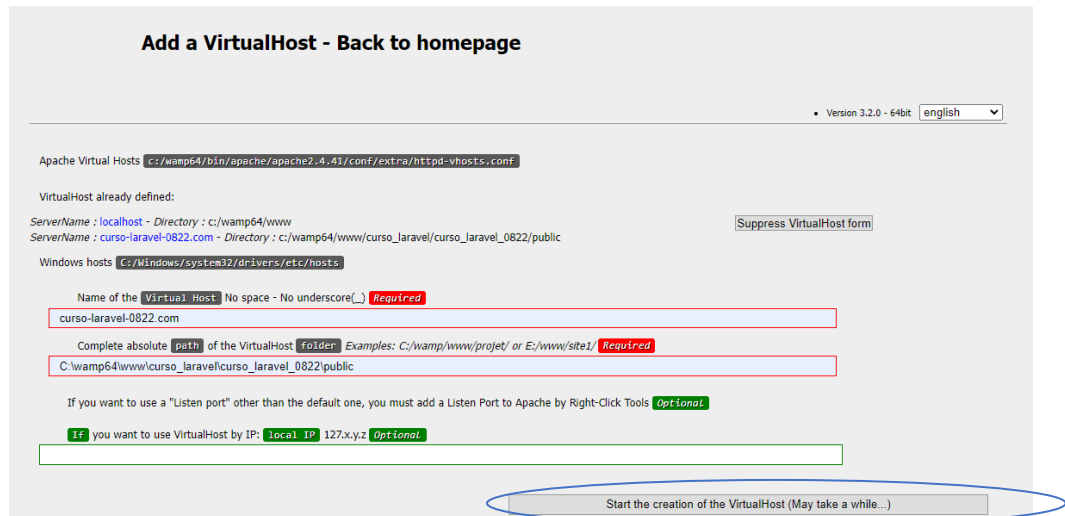
2.2 Usar comando `php artisan serv`, este comando es muy útil. Nos permite iniciar un servidor bajo un puerto determinado, y acceder a esa ruta para ver nuestro proyecto sin /public en la dirección.

Funciona incluso sin tener Apache instalado porque usa un servidor propio de PHP.

3. Creación Host Virtual; A través del este link <http://localhost/> se puede crear un VirtualHost

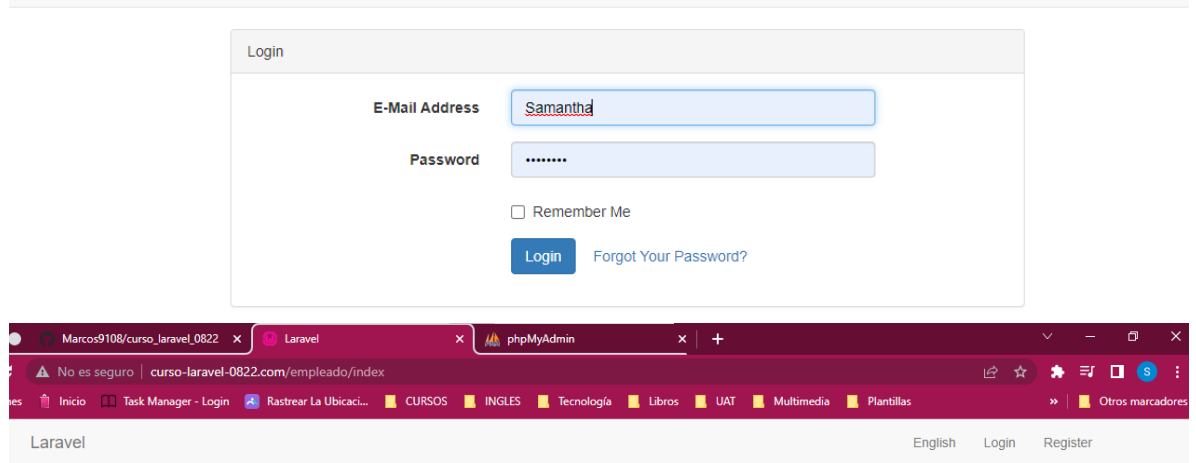
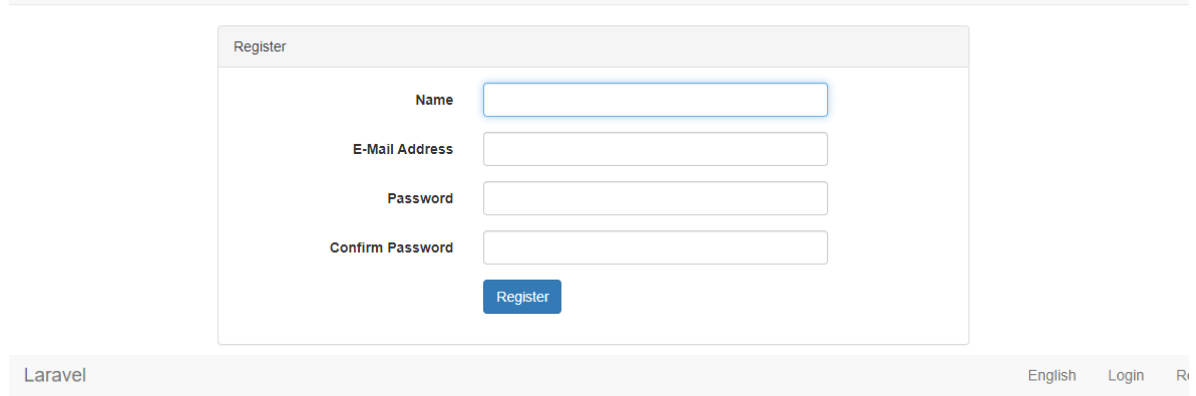
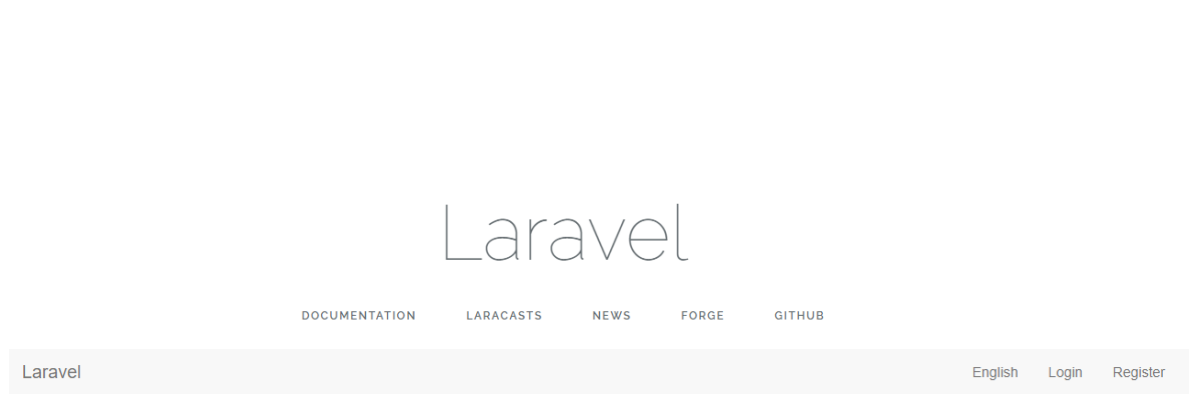
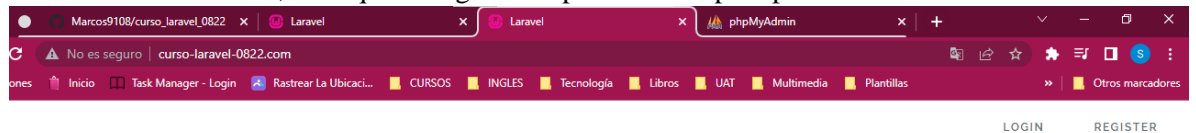


Damos clic en Agregar un Host Virtual. Una vez ingresado , agregaremos lo siguientes datos.



Finalizamos con Empezar con la creación del sitio virtual

4. Acceso a Virtual Host ; Se requiere registrarse previamente para poder acceder.



Listado empleados

Codigo empleado	Nombre	Apellidos	Correo	Acciones
90454	MARIA	Lopez Hernandez	marialo3@gmail.com	Eliminar

Creación de Login

1. Acceder a CMD, ejecutar la siguiente línea de comando; `php artisan make:auth`

```
Simbolo del sistema
Microsoft Windows [Versión 10.0.19043.1889]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Samantha>cd: C:\wamp64\www\curso_laravel\curso_laravel_0822
El nombre de archivo, el nombre de directorio o la sintaxis de la etiqueta del volumen no son correctos.
C:\Users\Samantha>cd C:\wamp64\www\curso_laravel\curso_laravel_0822

C:\wamp64\www\curso_laravel\curso_laravel_0822>php artisan make:auth

The [auth/login.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> no

The [auth/register.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> n

The [auth/passwords/email.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> [no]

The [auth/passwords/reset.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> no

The [layouts/app.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> NO

The [home.blade.php] view already exists. Do you want to replace it? (yes/no) [no]:
> NO

Authentication scaffolding generated successfully.

C:\wamp64\www\curso_laravel\curso_laravel_0822>
```

En este caso , ya lo tenía previamente cargado, pero estos son los archivos que ejecuta. Este comando permite instalar las rutas y las vistas necesarias para trabajar con el login, registro, recuperación de contraseñas y verificación de E-mail.

Creación de Migraciones

1. Para crear nuestras tablas en nuestra base de datos, debemos ejecutar el siguiente comando `php artisan migrate`

Estas son las migraciones que deberá descargar y estar en tu base de datos.

```
C:\wamp64\www\curso_laravel\curso_laravel_0822>php artisan migrate:status

+-----+-----+
| Ran? | Migration |
+-----+-----+
| Y     | 2014_10_12_000000_create_users_table |
| Y     | 2014_10_12_100000_create_password_resets_table |
| Y     | 2022_08_30_150722_create_empleado_table |
| Y     | 2022_08_31_134205_alter_empleado_table |
+-----+-----+

C:\wamp64\www\curso_laravel\curso_laravel_0822>
```

Servidor: MySQL:3308 » Base de datos: curso_laravel_0822 » Tabla: migrations

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios

Mostrando filas 0 - 3 (total de 4, La consulta tardó 0,0020 segundos.)

SELECT * FROM `migrations`

Perfilando [Editar en línea]

Mostrar todo Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: N

+ Opciones

		id	migration	batch
<input type="checkbox"/>	Editar	1	2014_10_12_000000_create_users_table	1
<input type="checkbox"/>	Editar	2	2014_10_12_100000_create_password_resets_table	1
<input type="checkbox"/>	Editar	3	2022_08_30_150722_create_empleado_table	1
<input type="checkbox"/>	Editar	4	2022_08_31_134205_alter_empleado_table	1

Seleccionar todo Para los elementos que están marcados: Editar Copiar Borrar Exportar

- Migración modelo empleados , utilizar el siguiente comando `php artisan make:model Empleado`

```
C:\wamp64\www\curso_laravel\curso_laravel_0822>php artisan make:model Empleado
Model already exists!
C:\wamp64\www\curso_laravel\curso_laravel_0822>
```

CREACION CRUD

UBICACIÓN; `app > Empleado.php`

Llenado de arrays, esto se realiza en la ubicación de nuestro código \$fillable en modelo Empleado.php.

```
EXPLORER
CURSO_LARAVEL_0822
  app
  Console
  Exceptions
  Http
  Providers
  Empleado.php
  User.php
  bootstrap
  config
  database
  public
  resources
  assets
  js
  sass
  lang
  views
  routes
  storage
  tests
  Feature
  Unit
  CreatesApplication.php
  TestCase.php
  vendor

app > Empleado.php
1 <?php
2
3 namespace App;
4
5 use Illuminate\Database\Eloquent\Model;
6
7 class Empleado extends Model
8 {
9     protected $table = 'empleado';
10
11     protected $fillable = ['id',
12         'nombre',
13         'apellido_paterno',
14         'apellido_materno',
15         'correo',
16         'fecha_nacimiento',
17         'ciudad',
18         'direccion',
19         'genero',
20         'telefono',
21         'codigo_empleado',
22         'puesto',
23         'edad',
24         'salario',
25         'tipo_moneda',
26         'activo',
27     ];
28 }
```

Se agregaron las líneas seleccionadas.

1. Crear Controlador para Empleado, con el siguiente comando `php artisan make:controller EmpleadoController --resource`

```
C:\wamp64\www\curso_laravel\curso_laravel_0822>php artisan make:controller EmpleadoController
Controller already exists!

C:\wamp64\www\curso_laravel\curso_laravel_0822>
```

2. En el archivo EmpleadoController, se agregaron las siguientes líneas de código.

A) Index

```
    /**
     * public function index()
     * {
     *     $empleados = Empleado::orderBy('id', 'DESC')->get();
     *     //dd($empleados);
     *     return view('Empleado.index', compact('empleados'));
     * }
```

B) Create

```
32 public function create()
33 {
34     return view('Empleado.create');
35 }
36
```

C) Store

```
EmpleadoController.php M X HomeController.php M 2014_10_12_000000_create_users_
app > Http > Controllers > EmpleadoController.php
37 /**
38  * Store a newly created resource in storage.
39  *
40  * @param \Illuminate\Http\Request $request
41  * @return \Illuminate\Http\Response
42  */
43 public function store(Request $request)
44 {
45     //dd($request->all());
46
47     $this->validate($request,[
48         'nombre' => 'required|max:10',
49         'apellido_paterno' => 'required',
50         'apellido_materno' => 'required',
51         'correo' => 'required|email', //Formato correo
52         'fecha-nacimiento' => '', //Solo acepte formato fecha
53         'ciudad' => 'required',
54         'direccion' => '',
55         'genero' => 'required', //Solo acepte masculino/femenino
56         'telefono' => 'required',
57         'codigo_empleado' => 'required', //Unico
58         'puesto' => 'required',
59         'edad' => 'required',
60         'salario' => 'required',
61         'tipo_moneda' => 'required',
62         'activo' => 'required'
63     ]);
64 }
```

D) ArraySave

```
EmpleadoController.php M X HomeController.php M 2014_10_12_000000
app > Http > Controllers > EmpleadoController.php
69         'correo' => 'required',
70         'fecha-nacimiento' => '',
71         'direccion' => '',
72         'genero' => 'required',
73         'telefono' => 'required',
74         'codigo_empleado' => 'required',
75         'puesto' => 'required',
76         'edad' => 'required',
77         'salario' => 'required',
78         'tipo_moneda' => 'required',
79         'activo' => 'required'
80     ]
81 );
82
83 dd($request->all(), $validaciones, $validaciones->errors);
84
85 $arraySave = [
86     'nombre' => $request->get("nombre"),
87     'apellido_paterno' => $request->get("apellido_paterno"),
88     'apellido_materno' => $request->get("apellido_materno"),
89     'correo' => $request->get("correo"),
90     'fecha-nacimiento' => $request->get("fecha_nacimiento"),
91     'ciudad' => $request->get("ciudad"),
92     'direccion' => $request->get("direccion"),
93     'genero' => $request->get("genero"),
94     'telefono' => $request->get("telefono"),
95     'codigo_empleado' => $request->get("codigo_empleado"),
96     'puesto' => $request->get("puesto"),
97     'edad' => $request->get("edad"),
98     'salario' => $request->get("salario"),
99     'tipo_moneda' => $request->get("tipo_moneda"),
100     'activo' => $request->get("activo")
101 ];
```

E) Show

```
116     public function show(Empleado $empleado)
117     {
118         $empleado = Empleado::find($id);
119         return view('Empleado.show', compact('empleado'));
120     }
```

F) Edit

```
128 public function edit(Empleado $empleado)
129 {
130     $empleado = Empleado::find($id);
131     return view('Empleado.edit', compact('empleado'));
132 }
```

G) Update

```
141 public function update(Request $request, $id)
142 {
143     $this->validate($request,[
144         'nombre' => 'required',
145         'apellido_paterno' => 'required',
146         'apellido_materno' => 'required',
147         'correo' => 'required|email',
148         'fecha-nacimiento' => '',
149         'ciudad' => 'required',
150         'direccion' => '',
151         'genero' => 'required', //Solo acepte masculino/femenino
152         'telefono' => 'required',
153         'codigo_empleado' => 'required', //Unico
154         'puesto' => 'required',
155         'edad' => 'required',
156         'salario' => 'required',
157         'tipo_moneda' => 'required',
158         'activo' => 'required',
159     ]);
160     Empleado::find($id)->update($request->all());
161
162     return redirect()->route('empleado.index')->with('success','Registro actualizado satisfactoriamente');
163 }
164
```

H) Destroy

```
172 public function destroy($id)
173 {
174
175     Empleado::find($id)->delete();
176     return redirect()->route('empleado.index')->with('success','Registro eliminado satisfactoriamente');
177 }
178
179
```

UBICACIÓN routes > web.php

3. Agregar ruta para las funciones del controlador en el archivo de rutas routes/web.php

Route::resource('empleado', 'EmpleadoController');

```
21
22 Route::get('/empleado/index', 'EmpleadoController@index')->name('empleado.index');
```

Creación de vistas

UBICACIÓN resources > views > layouts > app.blade.php

1. Creamos una nueva carpeta llamada layouts
2. Dentro de esta carpeta creamos el archivo layout.blade.php
3. Agregamos la plantilla blade.html


```

resources > views > layouts > app.blade.php
1 <!DOCTYPE html>
2 <html lang="{{ app()->getLocale() }}">
3 <head>
4     <meta charset="utf-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1">
7
8     <!-- CSRF Token -->
9     <meta name="csrf-token" content="{{ csrf_token() }}">
10
11     <title>{{ config('app.name', 'Laravel') }}</title>
12
13     <!-- Styles -->
14     <link href="{{ asset('css/app.css') }}" rel="stylesheet">
15     <!-- CDN JQuery -->
16     <script src="https://code.jquery.com/jquery-3.6.0.js"></script>
17     <script src="https://code.jquery.com/ui/1.13.2/jquery-ui.js"></script>
18
19     <!-- CDN Bootstrap -->
20     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
21
22     <script type='text/javascript'>
23         | var _CSRF_TOKEN = '{{ csrf_token() }}';
24     </script>
25
26     <!-- Scripts -->
27     <script type='text/javascript' src="{{ asset('js/functions.js') }}"></script>
28 </head>
29 <body>
30     <div id="app">
31         <nav class="navbar navbar-default navbar-static-top">
32             <div class="container">
33                 <div class="navbar-header">

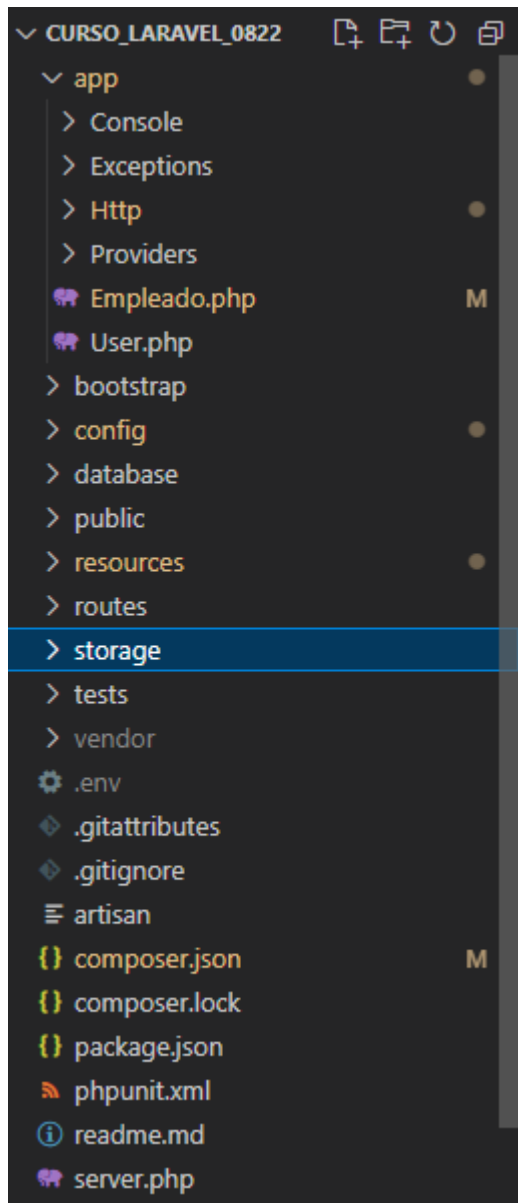
```

CRUD DATOS CONTACTO

app > Http > Controllers > DatoContactoController.php

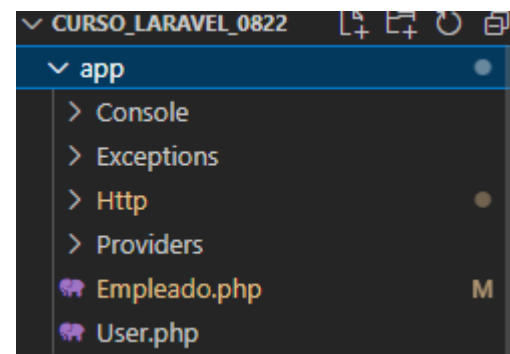
```
41     */
42     public function store(Request $request)
43     {
44         //dd($request->all());
45
46         $this->validate($request, [
47             'nombre_contacto' => 'required',
48             'email' => 'required|email',
49             'telefono' => 'required|numeric',
50             'direccion' => 'required',
51             'ciudad' => 'required',
52             'estado' => 'required',
53             'cp' => 'required'
54         ]);
55
56         $arraySave = [
57             'empleado_id' => $request->get("empleado_id"),
58             'nombre_contacto' => $request->get("nombre_contacto"),
59             'email' => $request->get('email'),
60             'telefono' => $request->get('telefono'),
61             'direccion' => $request->get('direccion'),
62             'ciudad' => $request->get('ciudad'),
63             'estado' => $request->get('estado'),
64             'cp' => $request->get('cp'),
65         ];
66
67         DatoContacto::create($arraySave);
68
69         return redirect()->route('empleado.show', $request->get('empleado_id'))->with('success', '');
70     }
71
72     /**
73      * Display the specified resource
```

Estructura proyecto

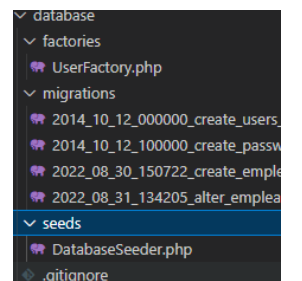
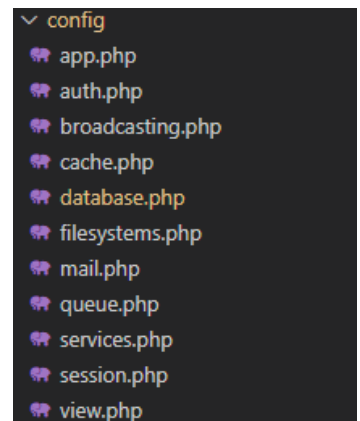
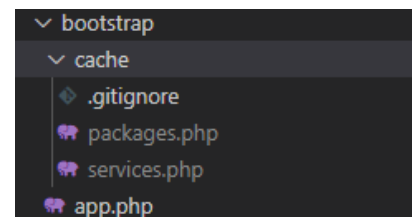


1. **app**: Contiene todo el código de la aplicación. Contiene toda la lógica de la aplicación, en este caso controladores, middlewares , el archivo kernel que se utiliza para configuracion
2. **app/Console**: Contiene todos los comandos artisan para usar en la aplicación.
3. **app/Exceptions**: Contiene las clases para manejar excepciones.
4. **app/Http**: Almacena los controladores creados en la aplicación.
5. **bootstrap**: Contiene los archivos y código que usa bootstrap.
6. **config**: Contiene los archivos de configuración.
7. **database**: Contiene las migraciones y archivos de creación de registros para pruebas.
8. **public**: Contiene archivos como imágenes, JavaScript CSS, etc.
9. **resources**: Ahí se encuentran todos los archivos y vistas de nuestra aplicación.
10. **routes**: Contiene las rutas para nuestra aplicación, (comunicación vista controlador).
11. **.env**: Este fichero se utiliza para almacenar los valores de configuración que son propios de la máquina o instalación actual.
12. **composer.json**: Este fichero es el utilizado por Composer para realizar la instalación de Laravel.

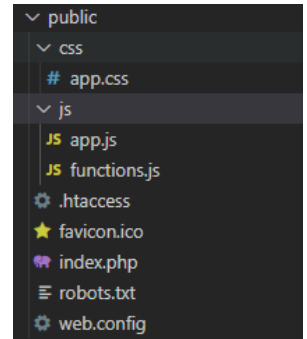
1. App
 - 1.1 Console
 - 1.1.1 Kernel.php
 - 1.2 Exceptions
 - 1.2.1 Handler.php
 - 1.3 Http
 - 1.3.1 Controllers
 - 1.3.1.1 Auth
 - 1.3.1.1.1 ForgotPasswordController.php
 - 1.3.1.1.2 LoginController.php
 - 1.3.1.1.3 RegisterController.php
 - 1.3.1.1.4 ResetPasswordController.php



- 1.3.1.2 Controller.php
 - 1.3.1.3 EmpleadoController.php
 - 1.3.1.4 HomeController.php
 - 1.3.2 Middleware
 - 1.3.2.1 EncryptCookies.php
 - 1.3.2.2 Language.php
 - 1.3.2.3 RedirectIfAuthenticated.php
 - 1.3.2.4 TrimStrings.php
 - 1.3.2.5 TrustProxies.php
 - 1.3.2.6 VerifyCsrfToken.php
 - 1.3.3 Kernel.php
- 1.4 Providers
 - 1.4.1 AppServiceProvider.php
 - 1.4.2 AuthServiceProvider.php
 - 1.4.3 BroadcastServiceProvider.php
 - 1.4.4 EventServiceProvider.php
 - 1.4.5 RouteServiceProvider.php
- 1.5 Empleado.php
- 1.6 User.php
- 2. Bootstrap
 - 2.1 Cache
 - 2.1.1 .gitignore
 - 2.1.2 packages.php
 - 2.1.3 services.php
 - 2.2 App.php
- 3. Config
 - 3.1 app.php
 - 3.2 auth.php
 - 3.3 broadcasting.php
 - 3.4 cache.php
 - 3.5 database.php
 - 3.6 filesystem.php
 - 3.7 mail.php
 - 3.8 queue.php
 - 3.9 services.php
 - 3.10 session.php
 - 3.11 view.php
- 4. Database
 - 4.1 factories
 - 4.1.1 UserFactory.php
 - 4.2 migrations
 - 4.2.1 2014_10_12_000000_create_users_table.php
 - 4.2.2 2014_10_12_100000_create_password_resets_table.php
 - 4.2.3 2022_08_30_150722_create_empleado_table.php
 - 4.2.4 2022_08_31_134205_alter_empleado_table.php
 - 4.3 seeds
 - 4.3.1 DatabaseSeeder.php



- 4.4 .gitignore
- 5. public
 - 5.1 css
 - 5.1.1 app.css
 - 5.2 js
 - 5.2.1 app.js
 - 5.2.2 functions.js
 - 5.3 .htaccess
 - 5.4 favicon.ico
 - 5.5 index.php
 - 5.6 robots.txt
 - 5.7 web.config
- 6. resources



NOTA. No se muestran las demás capturas por falta de tiempo, para subirlo exactamente a la hora indicada (11.59 como limite). Sin embargo en el código esta todo completo y funcional).