



**ESCUELA POLITÉCNICA NACIONAL**  
**ESCUELA DE FORMACIÓN DE TECNÓLOGOS**

---

**DESARROLLO DE APLICACIONES WEB**



ASIGNATURA:

Desarrollo de Aplicaciones Web

PROFESOR:

Ing. Loarte Byron

PERÍODO ACADÉMICO:

**LABORATORIO - 05**

**TÍTULO:**

**ESTRUCTURA EN LARAVEL**  
**RUTAS – CONTROLADORES – VISTAS**



## PROPÓSITO DE LA PRÁCTICA

Familiarizar al estudiante con la estructura de directorios en Laravel.

## OBJETIVO GENERAL

Determinar los principios en el desarrollo web con Laravel y su estructura de directorios.


## OBJETIVOS ESPECÍFICOS

- Configurar las herramientas para el laboratorio.
- Crear un proyecto en Laravel.
- Comprender los archivos y directorios más importantes en Laravel.
- Comprender el uso y manejo de Rutas.
- Comprender el uso y manejo de Controladores.
- Comprender el uso y manejo de Vistas.
- Comprender el uso y manejo del motor de plantillas Blade.
- Visualizar los resultados obtenidos.

## INSTRUCCIONES

1. Se procede con la creación de un nuevo proyecto en Laravel, para eso se debe ejecutar el siguiente comando en una terminal de Windows.

**laravel new concesionaria**



```
C:\Windows\System32\cmd.exe - laravel new concesionaria
Microsoft Windows [Versión 10.0.19042.1237]
(c) Microsoft Corporation. Todos los derechos reservados.
C:\Users\BYRONTOSH\Desktop\Introducción-PHP>laravel new concesionaria

Laravel

Creating a "laravel/laravel" project at "./concesionaria"
Installing laravel/laravel (v8.6.2)
- Installing laravel/laravel (v8.6.2): Extracting archive
Created project in C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
```

2. Luego de la creación del proyecto, se debe abrir el mismo en Visual Studio Code.

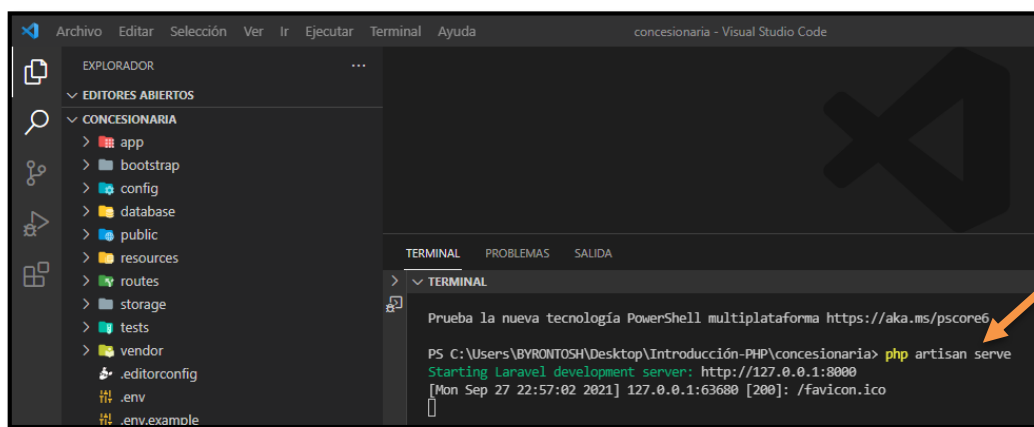
```
C:\Windows\System32\cmd.exe
Publishing complete.
> @php artisan key:generate --ansi
Application key set successfully.

Application ready! Build something amazing.

C:\Users\BYRONTOSH\Desktop\Introducción-PHP>cd concesionaria
C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>code .
```

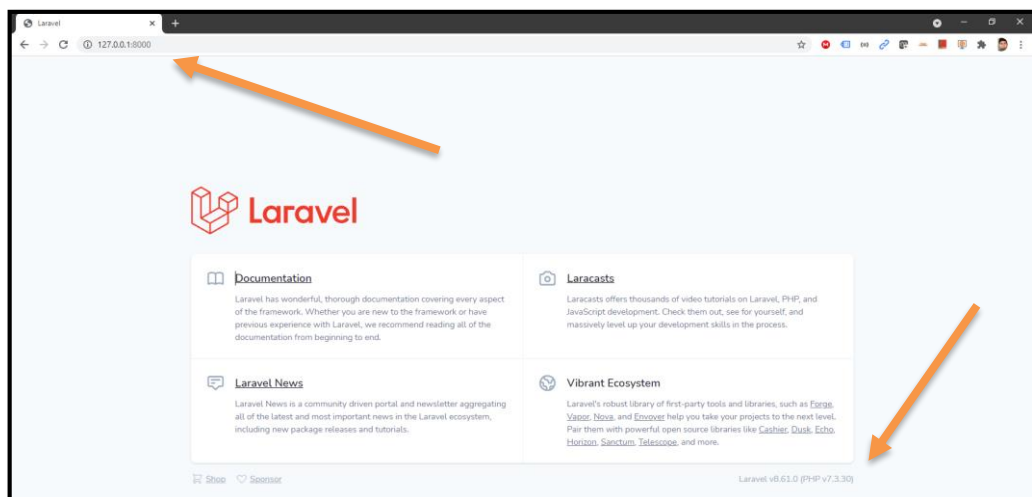
3. Luego se debe ejecutar el comando respectivo para ver el proyecto creado y ejecutado con éxito.

**php artisan serve**



4. Verificar el resultado obtenido

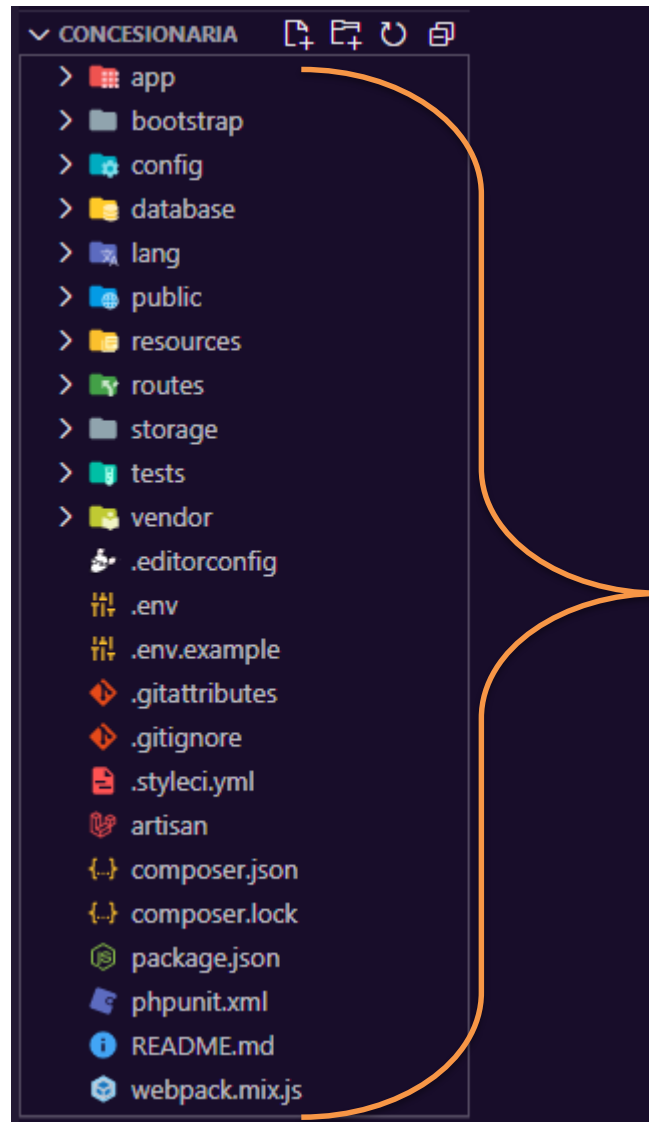
**<http://127.0.0.1:8000/>**



## ESTRUCTURA EN LARAVEL

5. Ahora se procede a ver la estructura de archivos y directorios en laravel y una pequeña descripción de cada uno de ellos.

<https://laravel.com/docs/9.x/structure>



## RUTAS



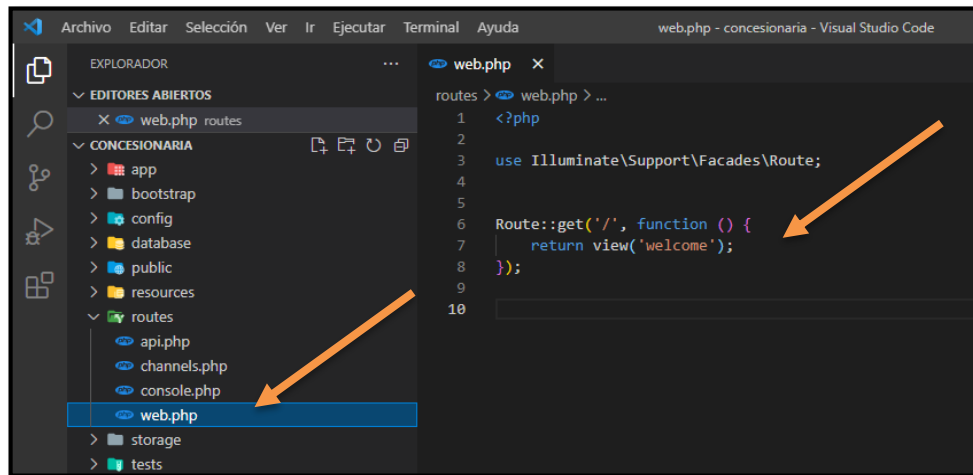
Las rutas son una capa muy importante en Laravel, es por ello que el Framework destina un directorio en la carpeta raíz, llamado **routes**, para ubicar todas las rutas de la aplicación.

Por defecto, tiene 4 archivos de rutas, pero las más relevantes son **web.php** y **api.php**.

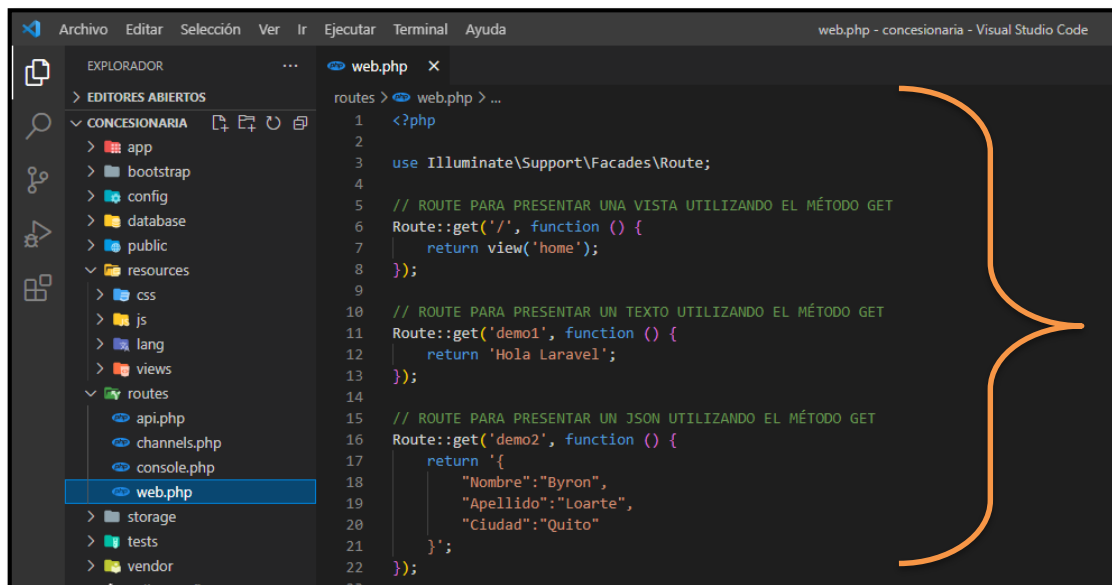
Como sus nombres lo expresan en **web.php** se definen las rutas para aplicaciones web y en **api.php** las rutas para crear APIs de tipo RESTFul.

<https://laravel.com/docs/9.x/routing>

- Ahora se procede a trabajar con rutas y sus diferentes variantes. En ese sentido abrir el archivo respectivo para crear las rutas correspondientes.



- Se procede con la creación de las diferentes rutas y la comprobación de cada una de ellas por medio de las vistas.



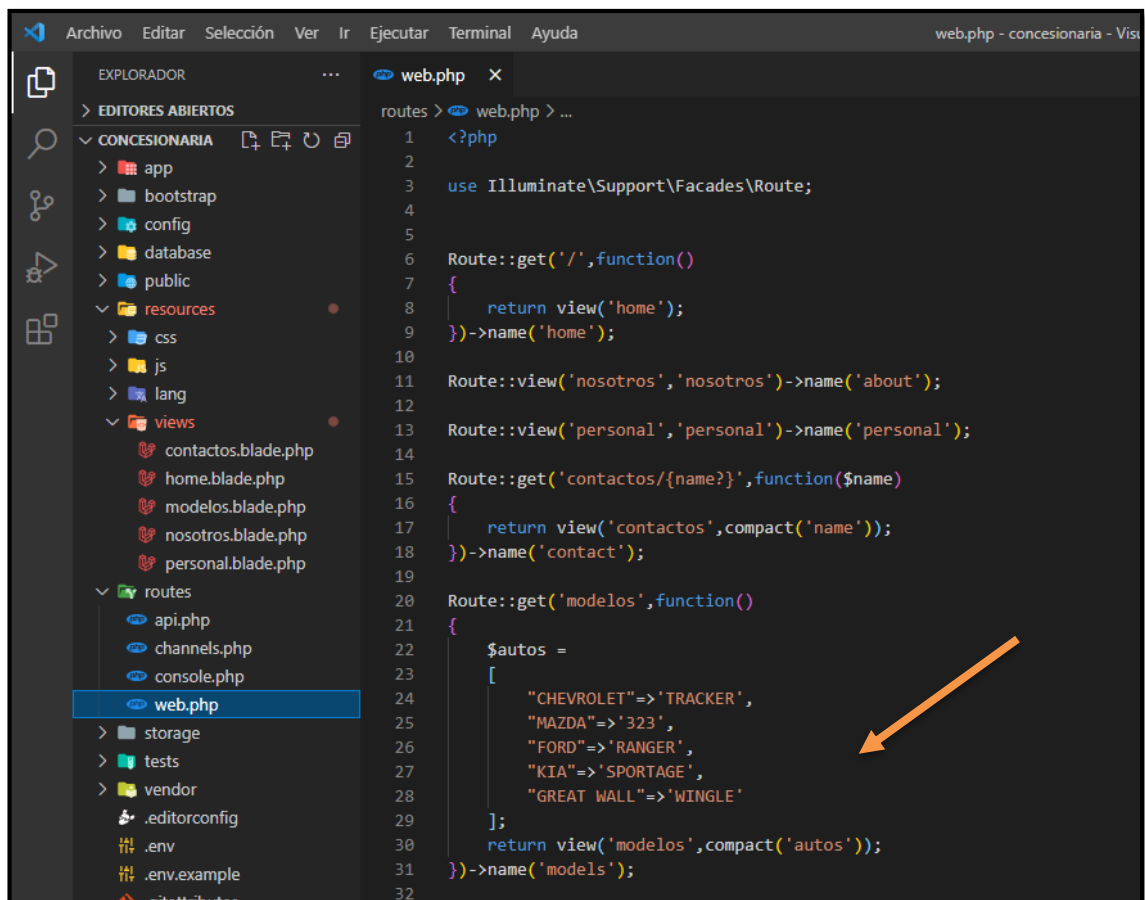
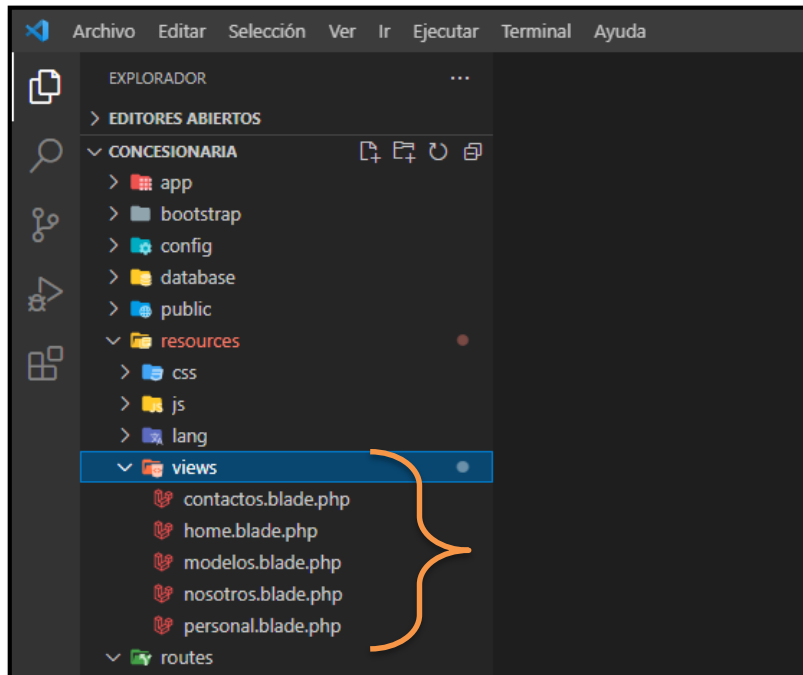
## MOTOR DE PLANTILLA BLADE



**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 usar código PHP simple en sus plantillas. De hecho, todas las plantillas de Blade se compilan en código PHP simple y se almacenan en caché hasta que se modifican, lo que significa que Blade agrega esencialmente cero gastos generales a su aplicación.

<https://laravel.com/docs/9.x/views>

8. Los archivos de plantilla Blade se declaran de la siguiente manera **autos.blade.php** y generalmente se almacenan en la siguiente ruta **resources/views**.

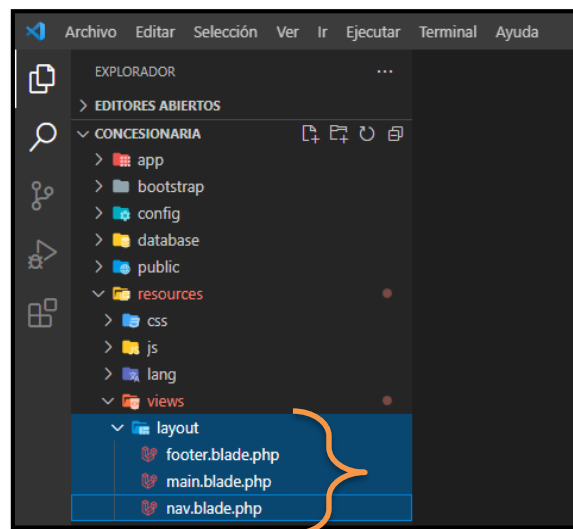


9. Ahora se procede a pasar datos a las vistas y presentar los mismos por medio de las directivas de **BLADE** y el uso de plantillas.

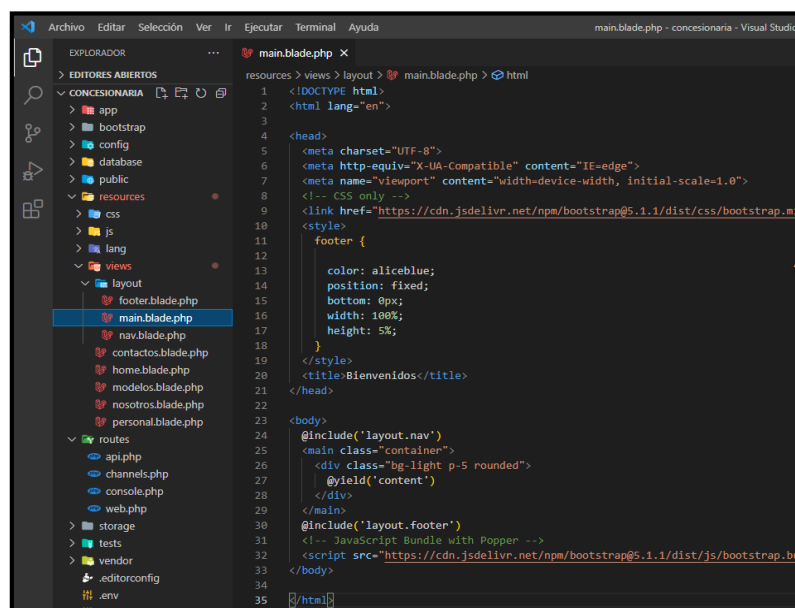


Además de la herencia de plantillas y la visualización de datos, Blade también proporciona accesos directos convenientes para estructuras de control PHP comunes, como declaraciones condicionales y bucles. Estos atajos proporcionan una forma muy clara y concisa de trabajar con las estructuras de control de PHP y, al mismo tiempo, siguen siendo familiares para sus contrapartes de PHP.

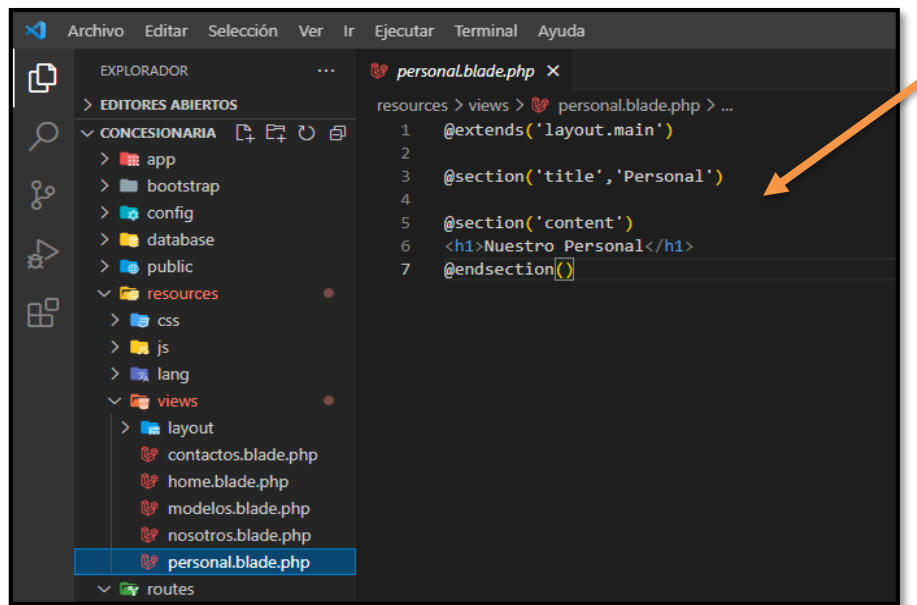
<https://laravel.com/docs/9.x/blade#introduction>



10. Posterior a ello se trabaja con el contenido de cada una de las dos vistas nav, footer y luego incluirlas en la vista main.

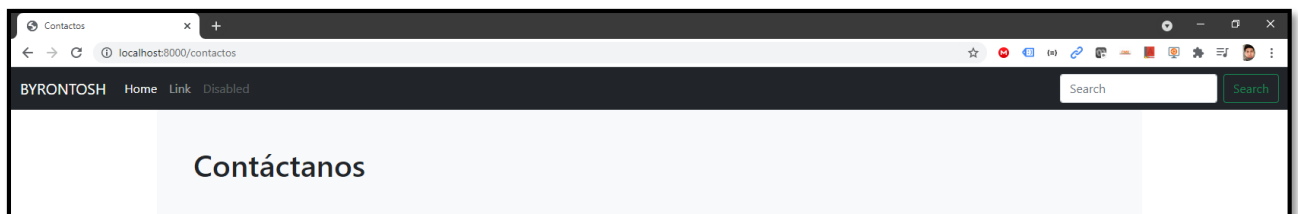


11. Ya con la plantilla establecida se procede a invocar en las otras páginas.

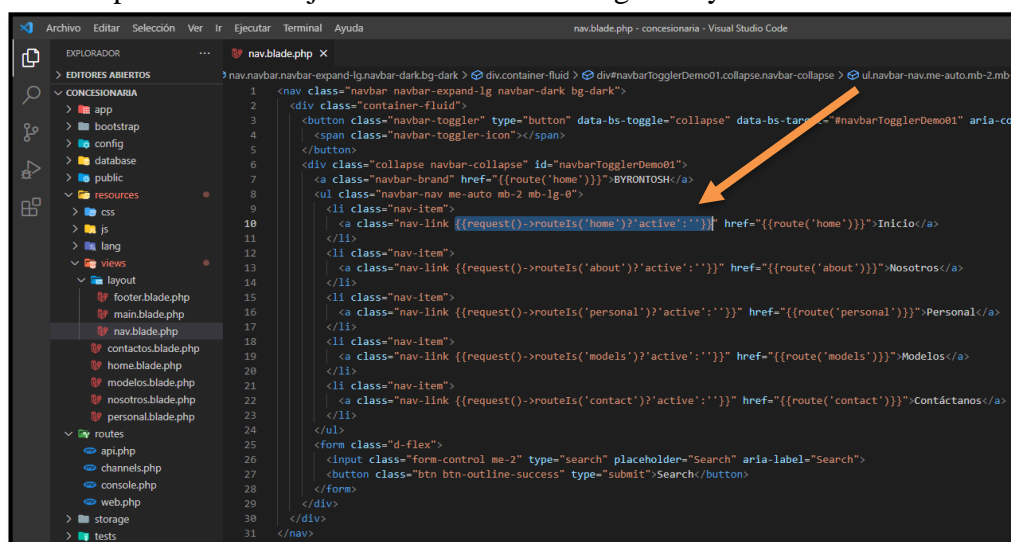


```
EXPLORADOR
> EDITORES ABIERTOS
CONCESIONARIA
  app
  bootstrap
  config
  database
  public
  resources
  css
  js
  lang
  views
    layout
      contactos.blade.php
      home.blade.php
      modelos.blade.php
      nosotros.blade.php
      personal.blade.php
  routes

personal.blade.php
resources > views > personal.blade.php > ...
1 @extends('layout.main')
2
3 @section('title', 'Personal')
4
5 @section('content')
6 <h1>Nuestro Personal</h1>
7 @endsection()
```



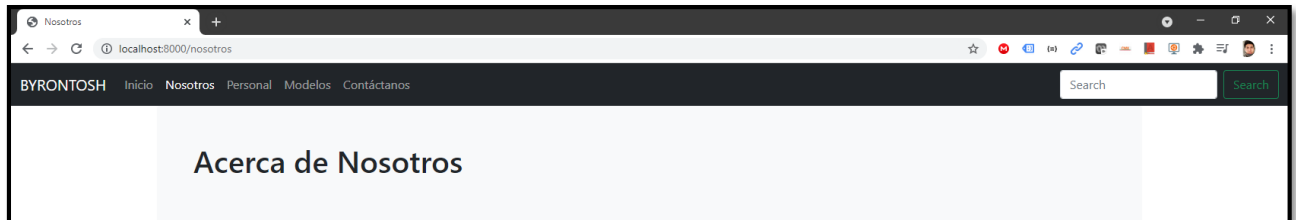
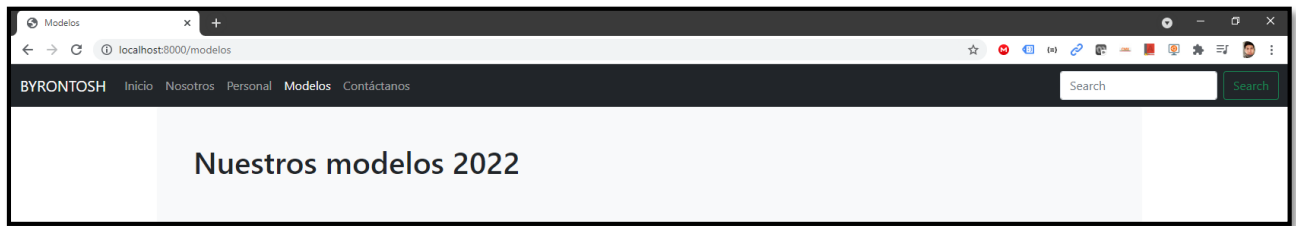
12. Ahora se procede a trabajar con el menú de navegación y las rutas



```
nav.blade.php - concesionaria - Visual Studio Code
EXPLORADOR
> EDITORES ABIERTOS
CONCESIONARIA
  app
  bootstrap
  config
  database
  public
  resources
  css
  js
  lang
  views
    footer.blade.php
    main.blade.php
    nav.blade.php
    contactos.blade.php
    home.blade.php
    modelos.blade.php
    nosotros.blade.php
    personal.blade.php
  routes
  api.php
  channels.php
  console.php
  web.php
  storage
  tests

nav.blade.php
1 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
2   <div class="container-fluid">
3     <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarTogglerDemo01" aria-co
4       <span class="navbar-toggler-icon"></span>
5     </button>
6     <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
7       <a class="navbar-brand" href="{{route('home')}}">BYRONTOSH</a>
8       <ul class="navbar-nav me-auto mb-2 mb-lg-0">
9         <li class="nav-item">
10          <a class="nav-link {{request()->routeIs('home')?'active':''}}" href="{{route('home')}}">Inicio</a>
11        </li>
12        <li class="nav-item">
13          <a class="nav-link {{request()->routeIs('about')?'active':''}}" href="{{route('about')}}">Nosotros</a>
14        </li>
15        <li class="nav-item">
16          <a class="nav-link {{request()->routeIs('personal')?'active':''}}" href="{{route('personal')}}">Personal</a>
17        </li>
18        <li class="nav-item">
19          <a class="nav-link {{request()->routeIs('models')?'active':''}}" href="{{route('models')}}">Modelos</a>
20        </li>
21        <li class="nav-item">
22          <a class="nav-link {{request()->routeIs('contact')?'active':''}}" href="{{route('contact')}}">Contáctanos</a>
23        </li>
24      </ul>
25      <form class="d-flex">
26        <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
27        <button class="btn btn-outline-success" type="submit">Search</button>
28      </form>
29    </div>
30  </div>
31 </nav>
```

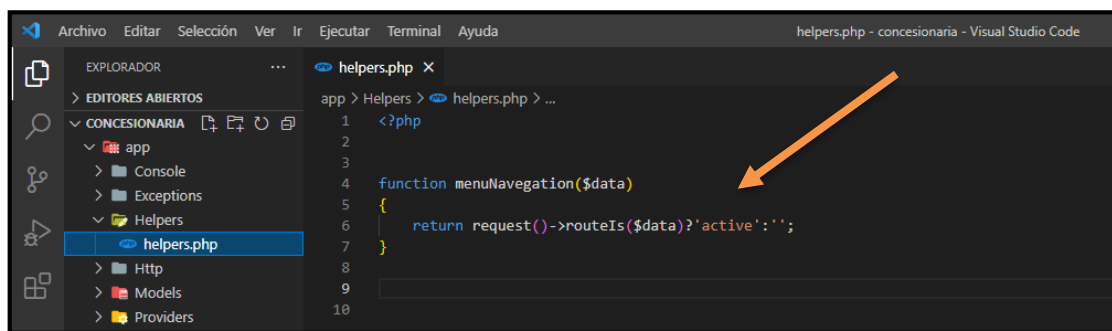




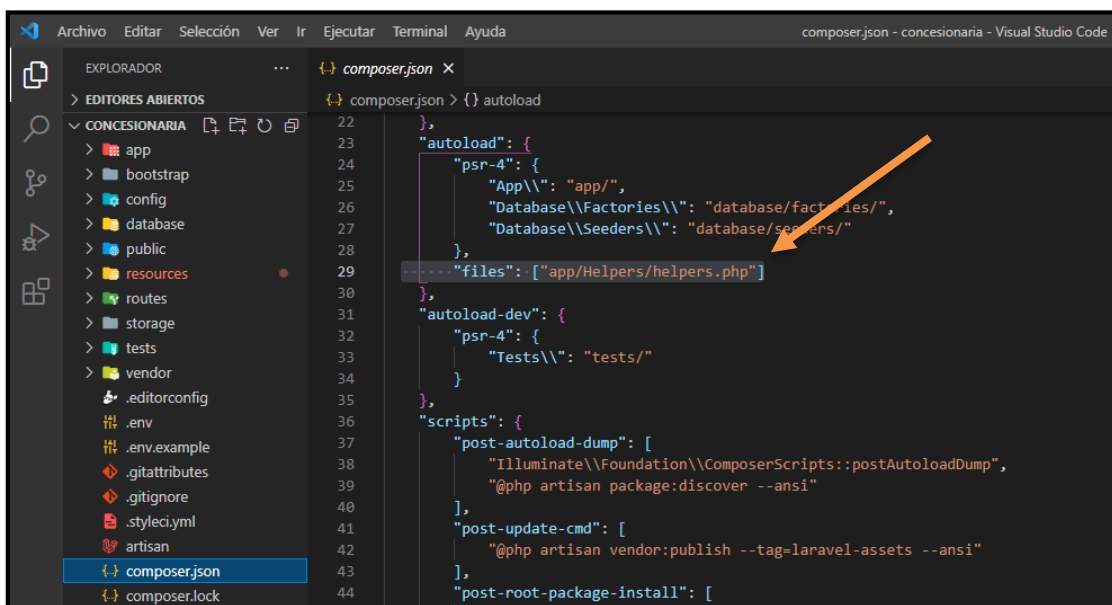
13. Se puede optimizar con el uso de los **Helpers** que provee Laravel.

<https://laravel.com/docs/9.x/helpers>

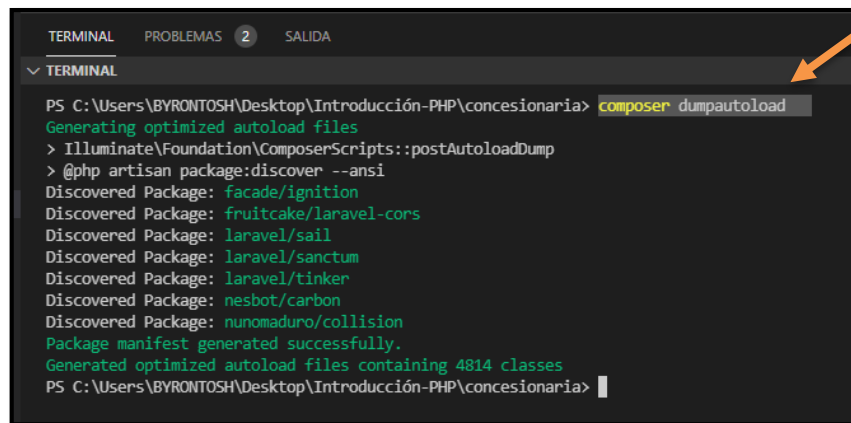
<https://laravel.com/docs/9.x/requests#inspecting-the-request-path>



14. Luego se procede a registrar el helper en el archivo composer.json

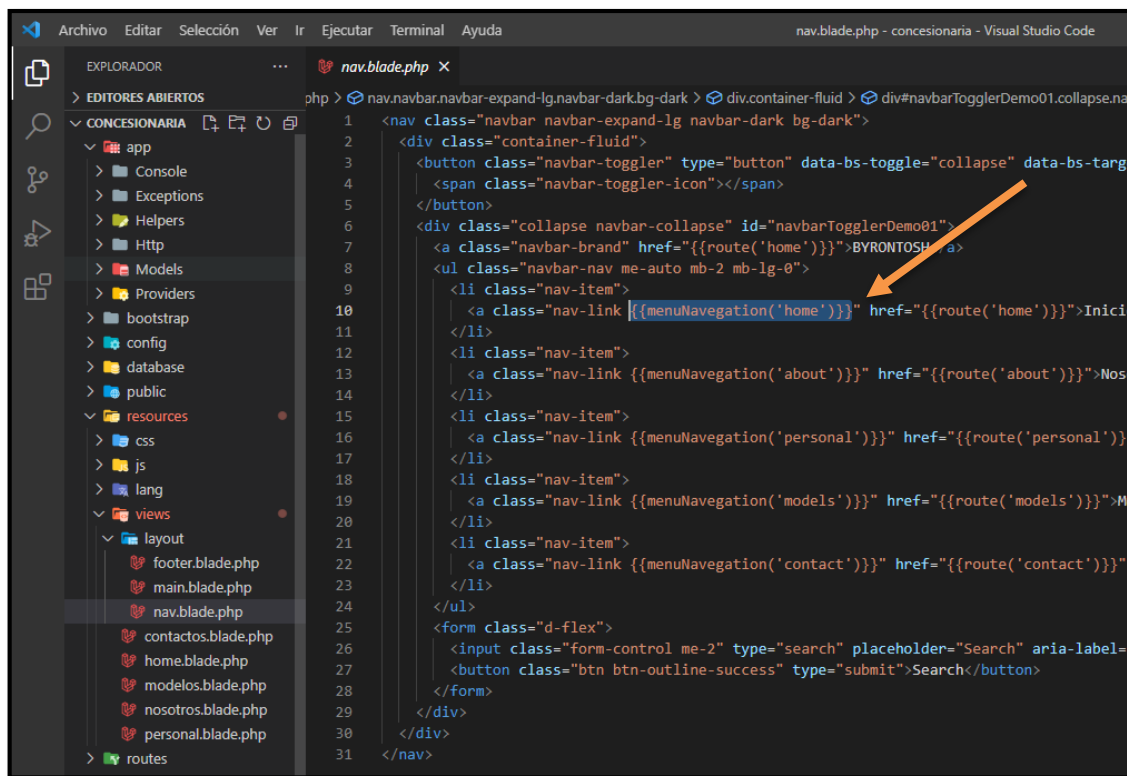


15. Ejecutar el comando **composer dumpautoload** en la terminal de comandos.

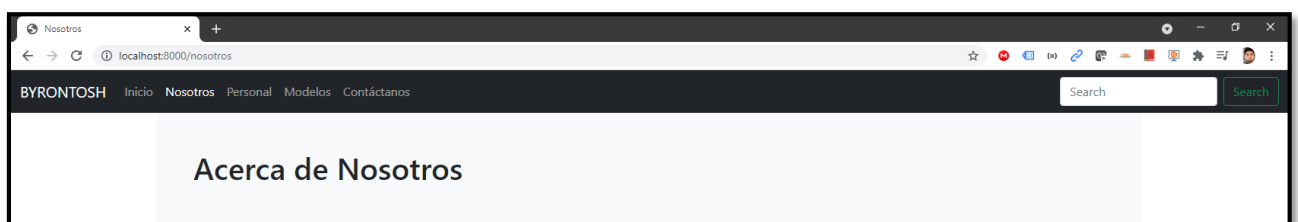
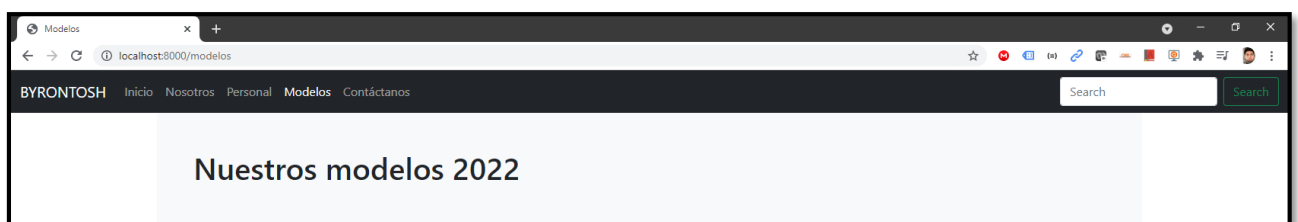


```
PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> composer dumpautoload
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fruitcake/laravel-cors
Discovered Package: laravel/sail
Discovered Package: laravel/sanctum
Discovered Package: laravel/tinker
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
Generated optimized autoload files containing 4814 classes
PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>
```

16. Invocar el método en la vista y verificar el resultado.

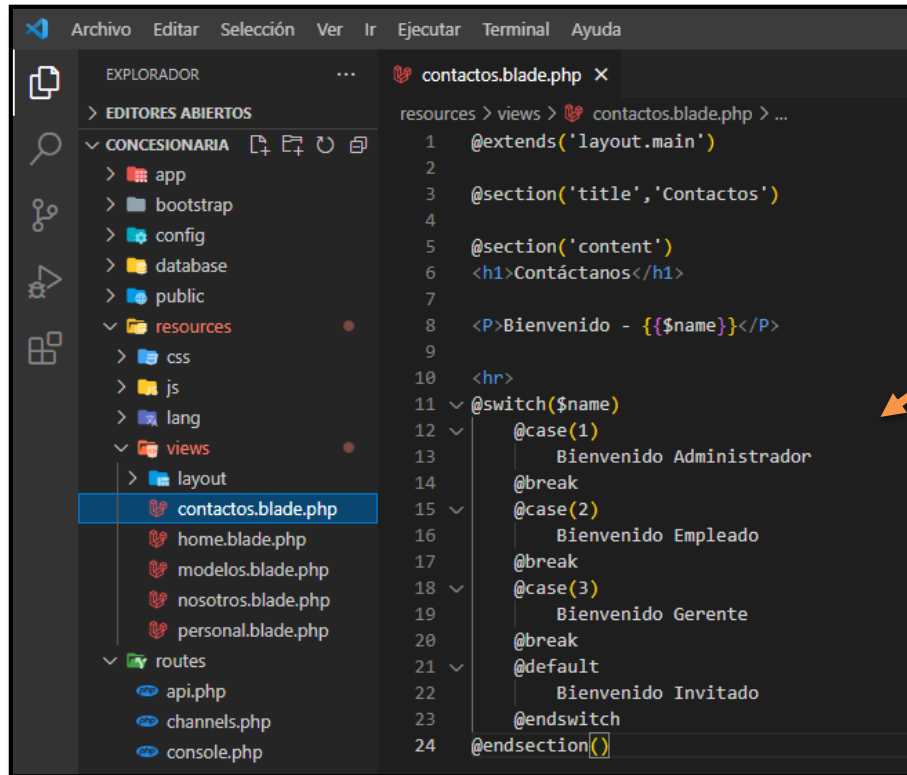


```
1 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
2   <div class="container-fluid">
3     <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarTogglerDemo01">
4       <span class="navbar-toggler-icon"></span>
5     </button>
6     <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
7       <a class="navbar-brand" href="{{route('home')}}">BYRONTOSH</a>
8       <ul class="navbar-nav me-auto mb-2 mb-lg-0">
9         <li class="nav-item">
10          <a class="nav-link" href="{{route('home')}}">Inicio
11        </li>
12        <li class="nav-item">
13          <a class="nav-link" href="{{route('about')}}">Nosotros
14        </li>
15        <li class="nav-item">
16          <a class="nav-link" href="{{route('personal')}}">Personal
17        </li>
18        <li class="nav-item">
19          <a class="nav-link" href="{{route('models')}}">Modelos
20        </li>
21        <li class="nav-item">
22          <a class="nav-link" href="{{route('contact')}}">Contactanos
23        </li>
24      </ul>
25    </div>
26    <form class="d-flex">
27      <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
28      <button class="btn btn-outline-success" type="submit">Search</button>
29    </form>
30  </div>
31 </nav>
```



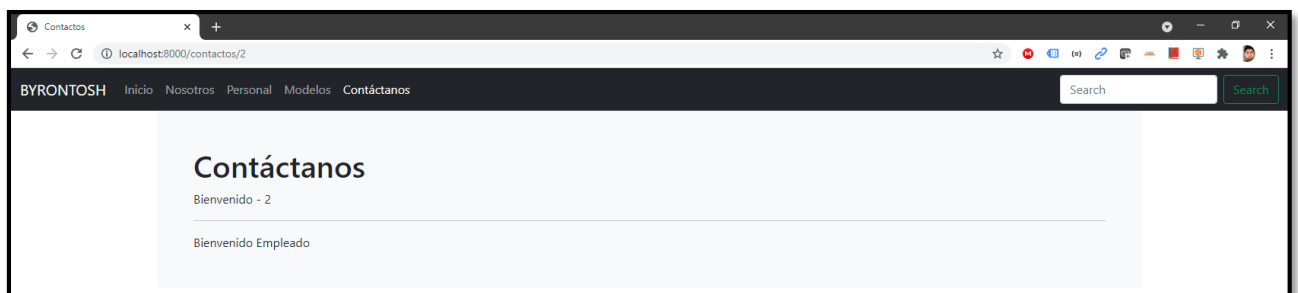
17. Además de la herencia de plantillas y la visualización de datos, Blade también proporciona accesos directos convenientes para estructuras de control PHP comunes, como declaraciones condicionales y bucles. Estos atajos proporcionan una forma muy clara y concisa de trabajar con las estructuras de control de PHP y, al mismo tiempo, siguen siendo familiares para sus contrapartes de PHP.

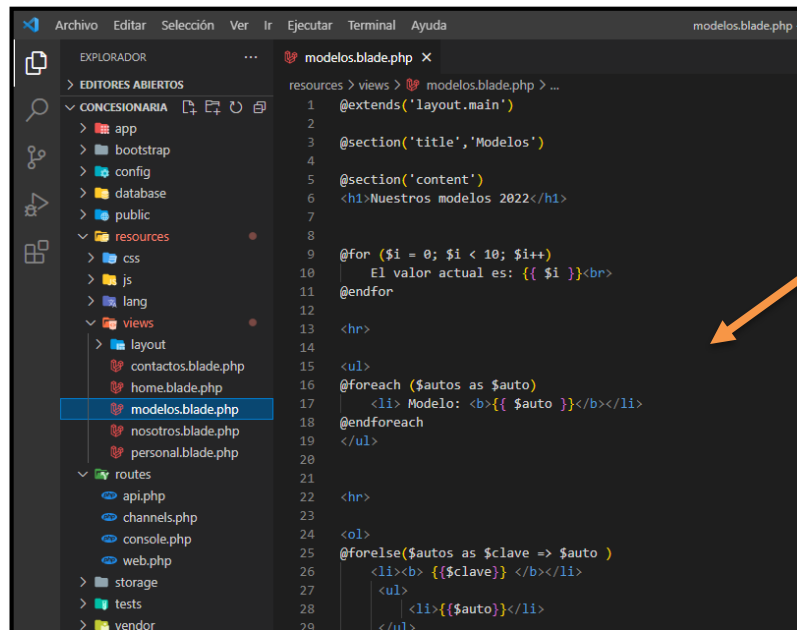
<https://laravel.com/docs/9.x/blade#blade-directives>



The screenshot shows the Visual Studio Code editor with the file explorer on the left and the code editor on the right. The file explorer shows the project structure with the 'views' folder expanded, and 'contactos.blade.php' selected. The code editor displays the content of 'contactos.blade.php' with line numbers 1 through 24. The code uses Blade directives to create a conditional welcome message based on the user's role. An orange arrow points to the @switch directive on line 11.

```
1 @extends('layout.main')
2
3 @section('title','Contactos')
4
5 @section('content')
6 <h1>Contáctanos</h1>
7
8 <P>Bienvenido - {{ $name }}</P>
9
10 <hr>
11 @switch($name)
12     @case(1)
13         Bienvenido Administrador
14     @break
15     @case(2)
16         Bienvenido Empleado
17     @break
18     @case(3)
19         Bienvenido Gerente
20     @break
21     @default
22         Bienvenido Invitado
23 @endswitch
24 @endsection()
```





```

1 @extends('layout.main')
2
3 @section('title', 'Modelos')
4
5 @section('content')
6 <h1>Nuestros modelos 2022</h1>
7
8
9 @for ($i = 0; $i < 10; $i++)
10     El valor actual es: {{ $i }}<br>
11 @endfor
12
13 <hr>
14
15 <ul>
16 @foreach ($autos as $auto)
17     <li> Modelo: <b>{{ $auto }}</b></li>
18 @endforeach
19 </ul>
20
21 <hr>
22
23 <ol>
24 @foreach($autos as $clave => $auto )
25     <li><b> {{ $clave }} </b></li>
26     <ul>
27         <li>{{ $auto }}</li>
28     </ul>
29 </li>

```



## CONTROLADORES



En lugar de definir toda su lógica de manejo de solicitudes como cierres en sus archivos de ruta, es posible que desee organizar este comportamiento utilizando clases de "controlador". Los controladores pueden agrupar la lógica de manejo de solicitudes relacionadas en una sola clase.

<https://laravel.com/docs/9.x/controllers>

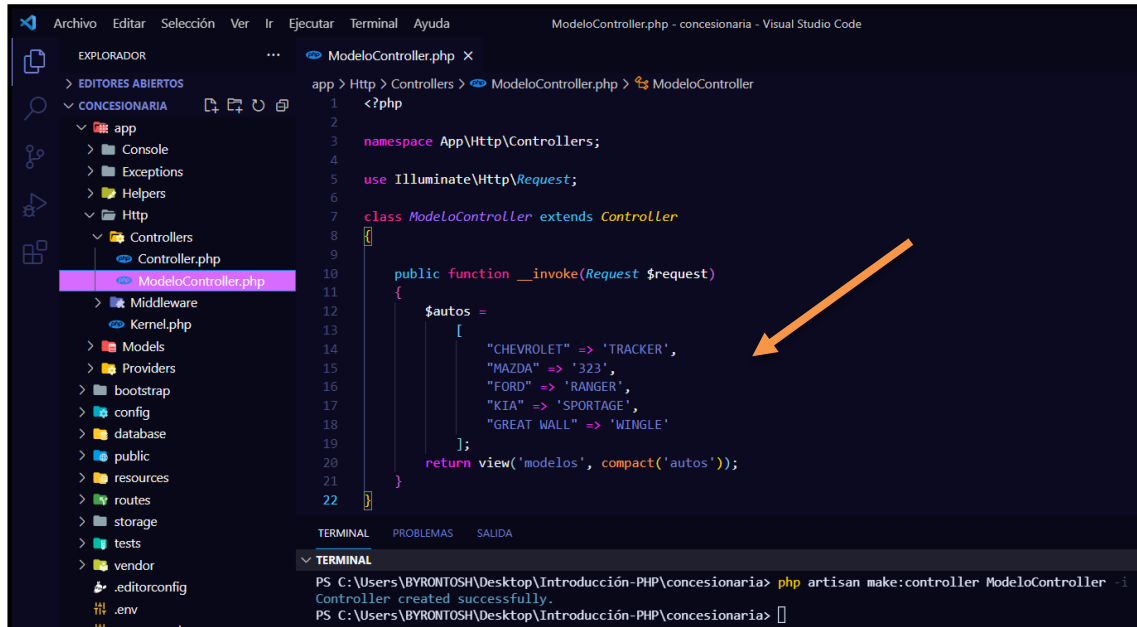
18. Por ejemplo, un **UserController** puede manejar todas las solicitudes entrantes relacionadas con los usuarios, lo que incluye mostrar, crear, actualizar y eliminar usuarios (CRUD).

De forma predeterminada, los controladores se almacenan en el directorio **app/Http/Controllers**.

19. Se procede con la creación de un nuevo controlador con la opción `-i`, el cual provee un controlador de acción simple, el cual permite tener una sola acción que procesar.

**php artisan make:controller ModeloController -i**

<https://laravel.com/docs/9.x/controllers#single-action-controllers>

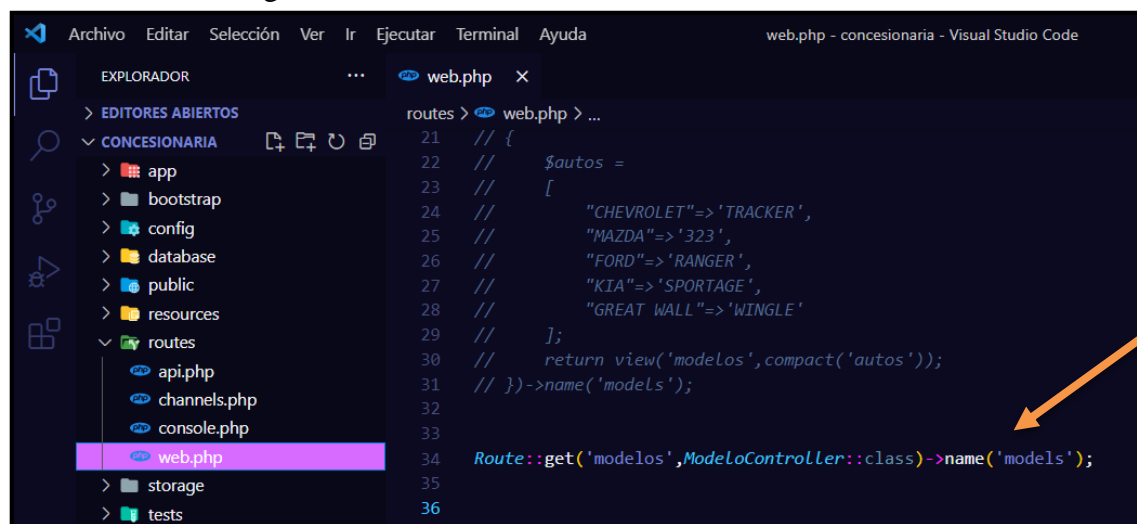


```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6
7 class ModeloController extends Controller
8 {
9
10     public function __invoke(Request $request)
11     {
12         $autos =
13         [
14             "CHEVROLET" => 'TRACKER',
15             "MAZDA" => '323',
16             "FORD" => 'RANGER',
17             "KIA" => 'SPORTAGE',
18             "GREAT WALL" => 'WINGLE'
19         ];
20         return view('modelos', compact('autos'));
21     }
22 }
```

TERMINAL

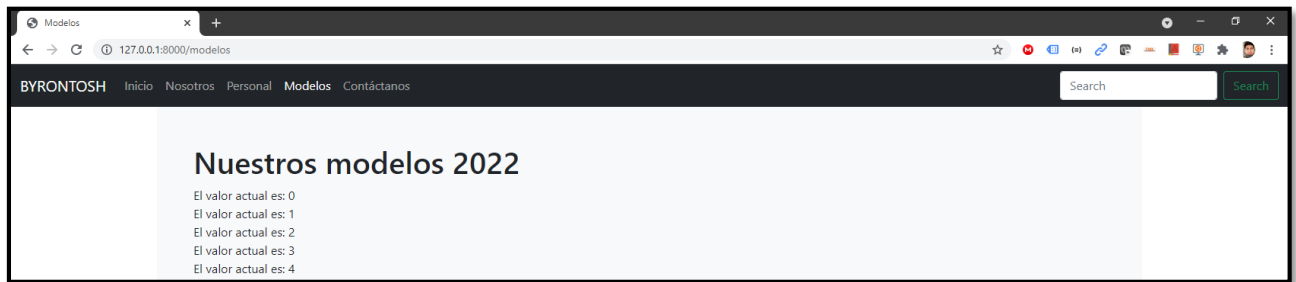
```
PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> php artisan make:controller ModeloController -i
Controller created successfully.
PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>
```

20. Al registrar rutas para controladores de **acción única**, no es necesario especificar un método de controlador. En su lugar, simplemente puede pasar el nombre del controlador a la ruta, de la siguiente manera:



```
21 // {
22 //     $autos =
23 //     [
24 //         "CHEVROLET"=>'TRACKER',
25 //         "MAZDA"=>'323',
26 //         "FORD"=>'RANGER',
27 //         "KIA"=>'SPORTAGE',
28 //         "GREAT WALL"=>'WINGLE'
29 //     ];
30 //     return view('modelos',compact('autos'));
31 // }->name('models');
32
33
34 Route::get('modelos',ModeloController::class)->name('models');
35
36
```

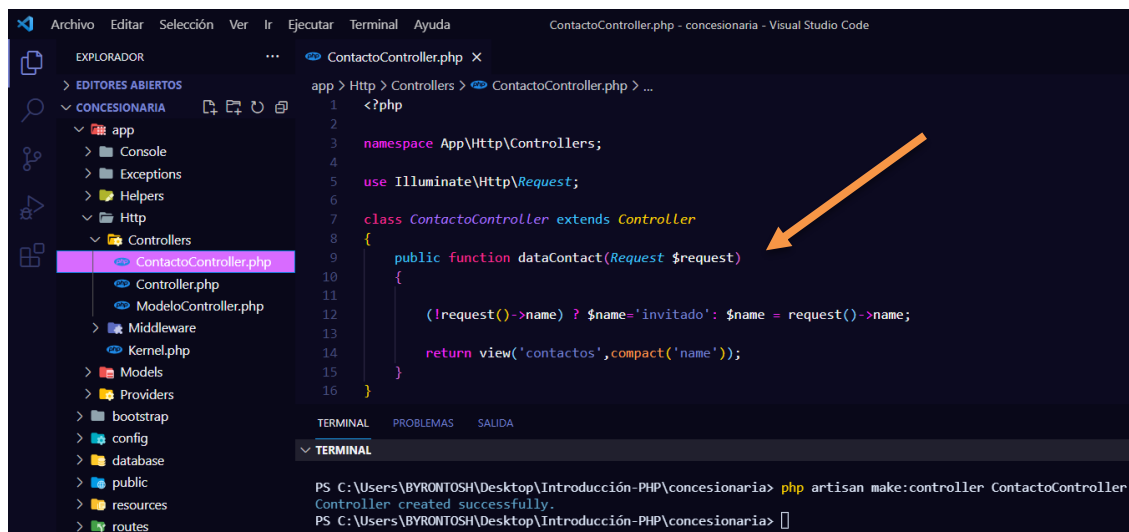
21. De esa manera toda la lógica se maneja en el controlador directamente.



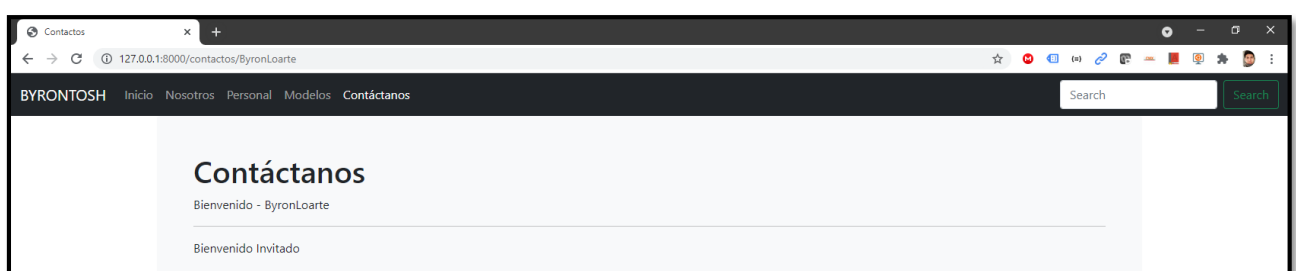
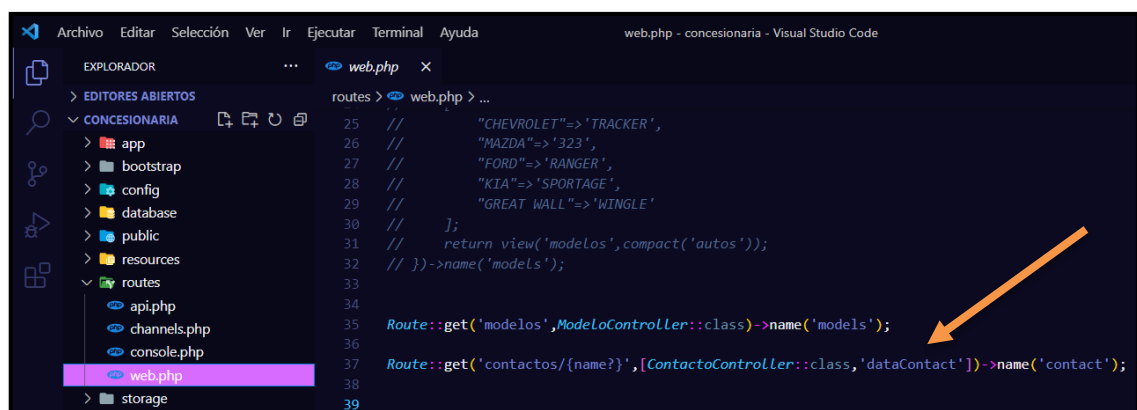
22. De la misma manera se puede crear controladores con métodos personalizados.

**php artisan make:controller ContactoController**

<https://laravel.com/docs/9.x/controllers#writing-controllers>



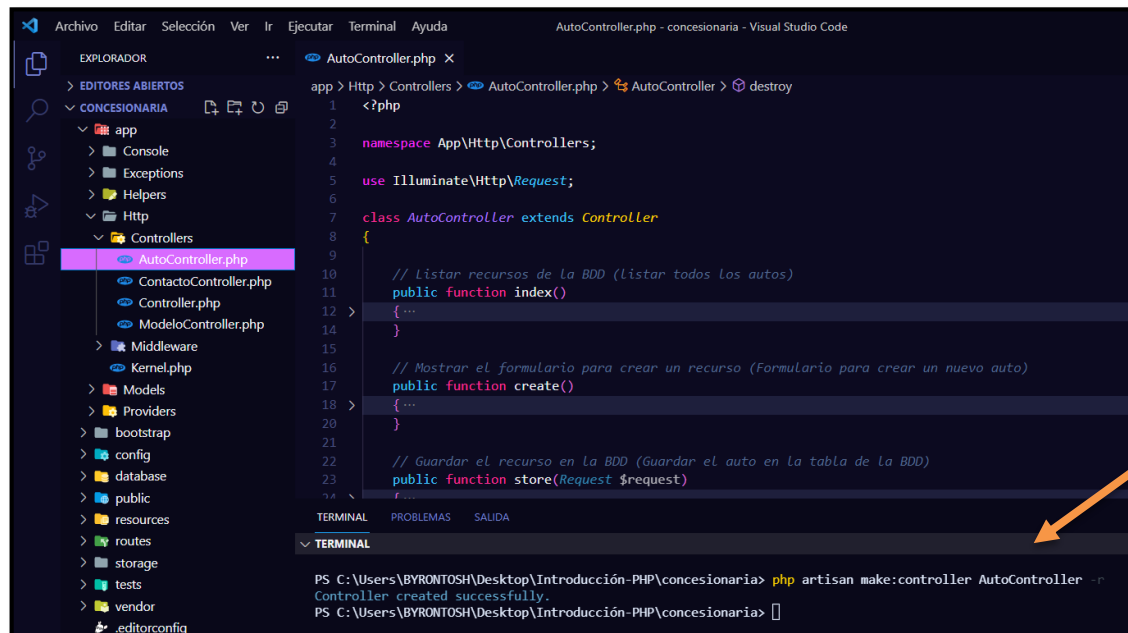
23. Se procede con la invocación del controlador y su método respectivo.



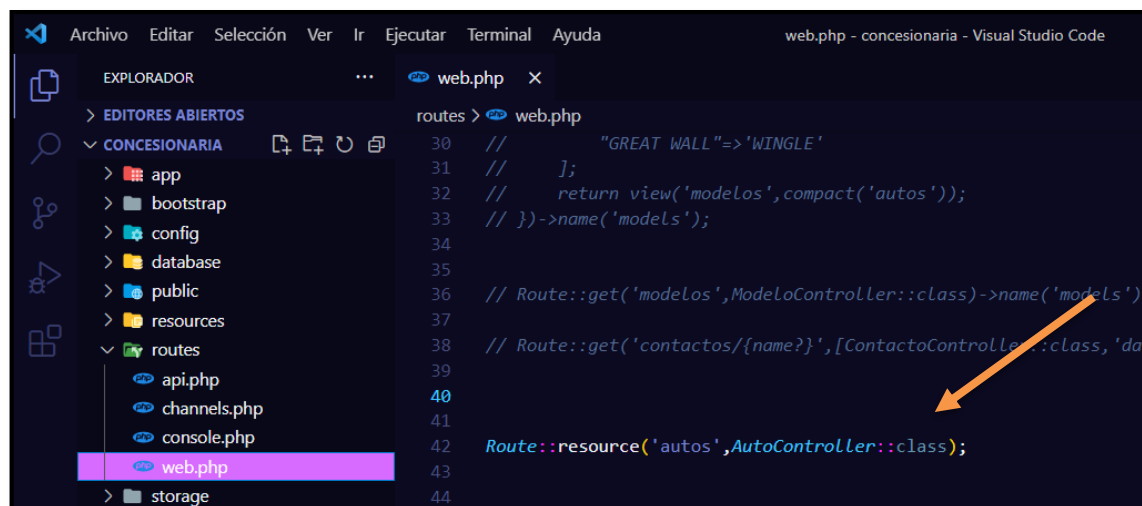
24. Ahora se procede a crear un tipo de **controlador de recursos**. Si piensa en cada modelo de Eloquent en su aplicación como un "recurso", es típico realizar los mismos conjuntos de acciones contra cada recurso en su aplicación. Por ejemplo, imagine que su aplicación contiene un modelo Photo y un modelo Movie. Es probable que los usuarios puedan crear, leer, actualizar o eliminar estos recursos.

<https://laravel.com/docs/9.x/controllers#resource-controllers>

**php artisan make:controller AutoController -r**



25. Ahora se procede a invocar el controlador en las rutas.



26. Se puede inspeccionar las rutas con el siguiente comando.

```

40 Route::resource('autos',AutoController::class);
41
42

```

TERMINAL PROBLEMAS SALIDA

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> php artisan route:list

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	api/user		Closure	api
	POST	autos	autos.index	App\Http\Controllers\AutoController@index	App\Http\Middleware\Authenticate:sanctum
	GET HEAD	autos/create	autos.store	App\Http\Controllers\AutoController@store	web
	GET HEAD	autos/{auto}	autos.create	App\Http\Controllers\AutoController@create	web
	PUT PATCH	autos/{auto}	autos.show	App\Http\Controllers\AutoController@show	web
	DELETE	autos/{auto}	autos.update	App\Http\Controllers\AutoController@update	web
	GET HEAD	autos/{auto}/edit	autos.destroy	App\Http\Controllers\AutoController@destroy	web
	GET HEAD	sanctum/csrf-cookie	autos.edit	App\Http\Controllers\AutoController@edit	web
				Laravel\Sanctum\Http\Controllers\CsrfCookieController@show	web

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>

27. Al declarar una ruta de recursos, se puede especificar un subconjunto de acciones que el controlador debe manejar en lugar del conjunto completo de acciones predeterminadas.

```

40
41 Route::resource('autos',AutoController::class)->only('index','show');
42

```

TERMINAL PROBLEMAS SALIDA

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> php artisan route:list

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	api/user		Closure	api
	GET HEAD	autos	autos.index	App\Http\Controllers\AutoController@index	App\Http\Middleware\Authenticate:sanctum
	GET HEAD	autos/{auto}	autos.show	App\Http\Controllers\AutoController@show	web
	GET HEAD	sanctum/csrf-cookie		Laravel\Sanctum\Http\Controllers\CsrfCookieController@show	web

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>

```

40
41 Route::resource('autos',AutoController::class)->except(['index','show']);
42

```

TERMINAL PROBLEMAS SALIDA

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> php artisan route:list

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	api/user		Closure	api
	POST	autos	autos.store	App\Http\Controllers\AutoController@store	App\Http\Middleware\Authenticate:sanctum
	GET HEAD	autos/create	autos.create	App\Http\Controllers\AutoController@create	web
	PUT PATCH	autos/{auto}	autos.update	App\Http\Controllers\AutoController@update	web
	DELETE	autos/{auto}	autos.destroy	App\Http\Controllers\AutoController@destroy	web
	GET HEAD	autos/{auto}/edit	autos.edit	App\Http\Controllers\AutoController@edit	web
	GET HEAD	sanctum/csrf-cookie		Laravel\Sanctum\Http\Controllers\CsrfCookieController@show	web

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>

```

42
43 Route::get('autos','App\Http\Controllers\AutoController@index')->name('allstore');
44
45 Route::delete('autos','App\Http\Controllers\AutoController@destroy')->name('destroystore');
46
47

```

TERMINAL PROBLEMAS SALIDA

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria> php artisan route:list

Domain	Method	URI	Name	Action	Middleware
	GET HEAD	api/user		Closure	api
	GET HEAD	autos	allstore	App\Http\Controllers\AutoController@index	App\Http\Middleware\Authenticate:sanctum
	DELETE	autos	destroystore	App\Http\Controllers\AutoController@destroy	web
	GET HEAD	sanctum/csrf-cookie		Laravel\Sanctum\Http\Controllers\CsrfCookieController@show	web

PS C:\Users\BYRONTOSH\Desktop\Introducción-PHP\concesionaria>

28. Ahora se procede a subir el proyecto a un repositorio.

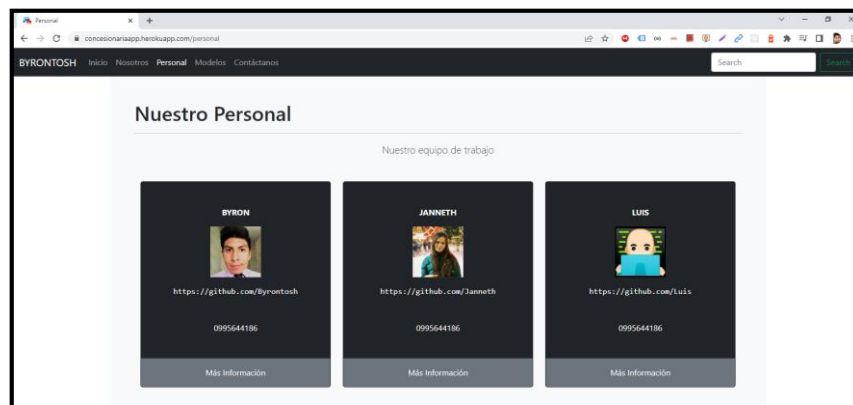
[https://github.com/Byrontosh/01\\_Concesionaria](https://github.com/Byrontosh/01_Concesionaria)



## TAREA DESAFÍO

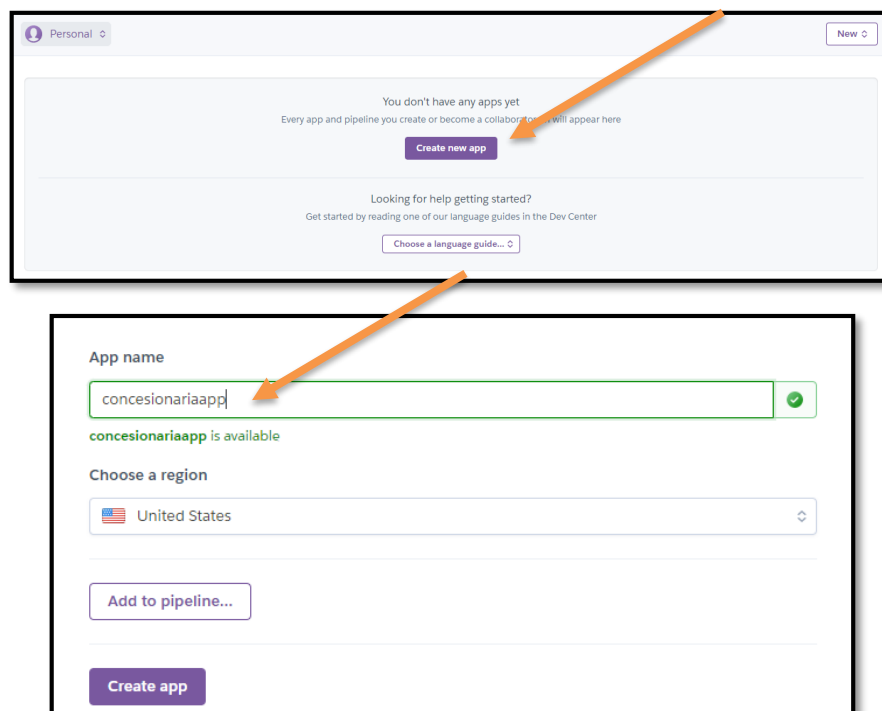


1. Completar cada una de las páginas del sistema web con cualquier contenido, pueden agregar más páginas si lo desean.

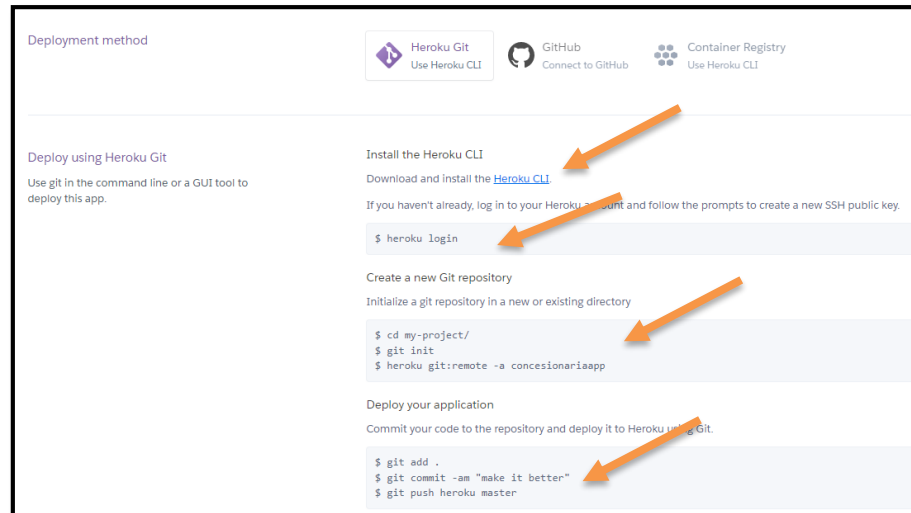
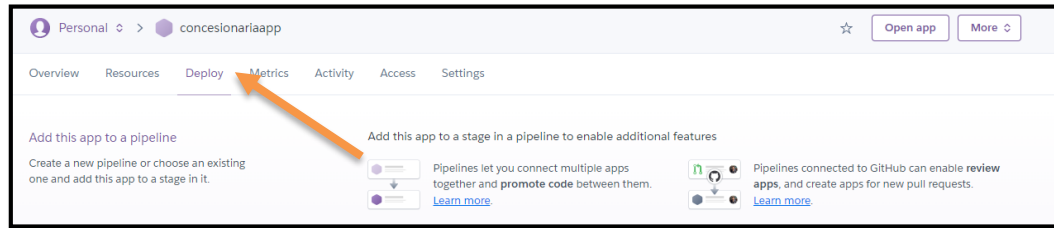


2. Realizar el despliegue a Heroku del sistema web, siguiendo los siguientes pasos que se describen a continuación.

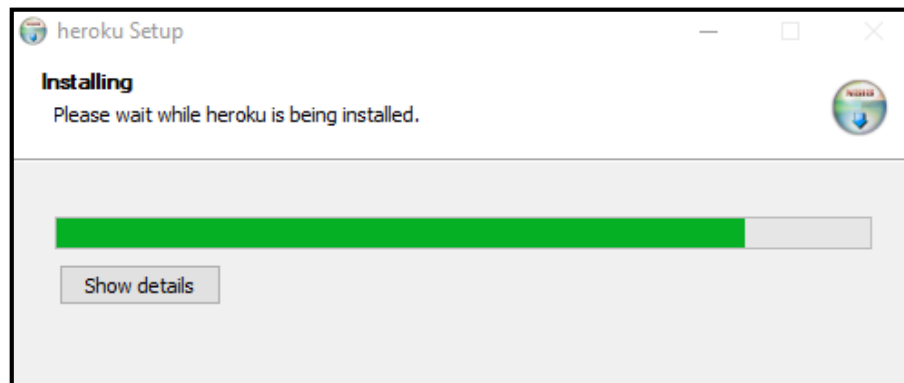
Crear una nueva aplicación



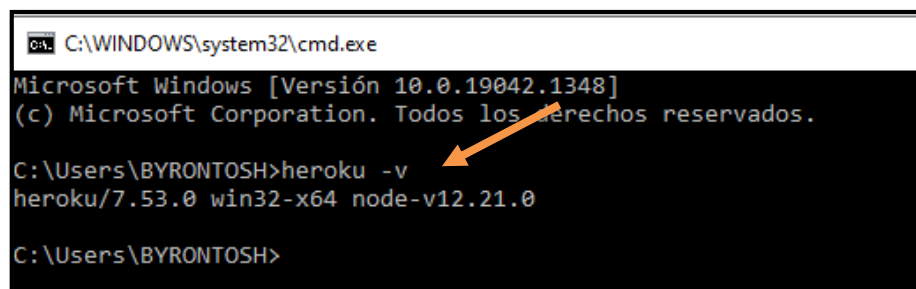
Luego dirigirse al apartado de Deploy y ejecutar los pasos requeridos



Para lo cual se procede a descargar e instalar Heroku CLI



Verificar si se instalo

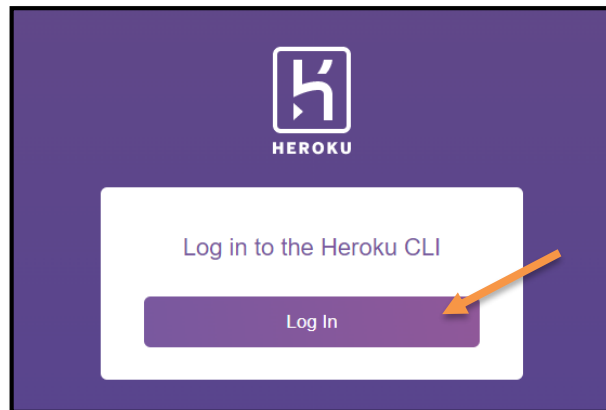


El siguiente paso es ejecutar heroku login

```
C:\> Seleccionar C:\WINDOWS\system32\cmd.exe - heroku login
Microsoft Windows [Versión 10.0.19042.1348]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\BYRONTOSH>heroku -v
heroku/7.53.0 win32-x64 node-v12.21.0

C:\Users\BYRONTOSH>heroku login
heroku: Press any key to open up the browser to login or q to exit:
```



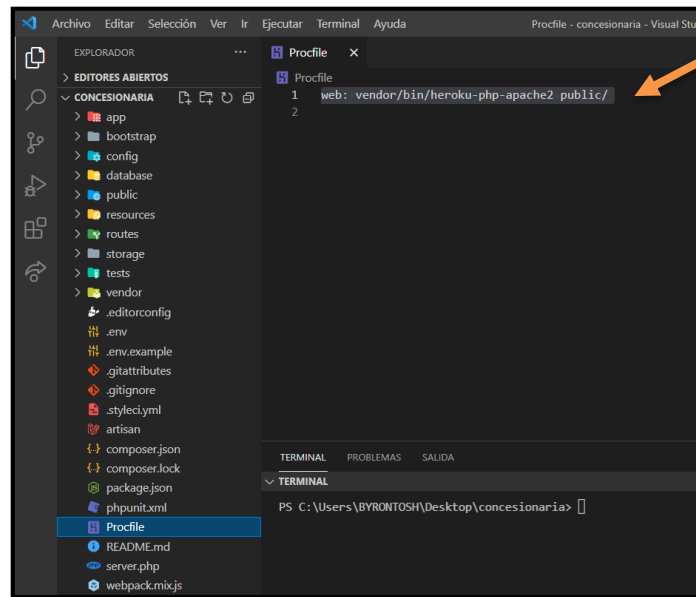
```
C:\> C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.19042.1348]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\BYRONTOSH>heroku -v
heroku/7.53.0 win32-x64 node-v12.21.0

C:\Users\BYRONTOSH>heroku login
heroku: Press any key to open up the browser to login or q to
Opening browser to https://cli-auth.heroku.com/auth/cli/browse
2gDbQAAAA4xNTcuMTAwLjE3MC4xOW4GANj3sjB9A1IAAVGA.qs1AidPERcZ4i
Logging in... done
Logged in as by_tosh20@hotmail.com

C:\Users\BYRONTOSH>
```

Ahora se procede a crear un archivo llamado Procfile con el siguiente contenido



Ejecutar los siguientes comandos

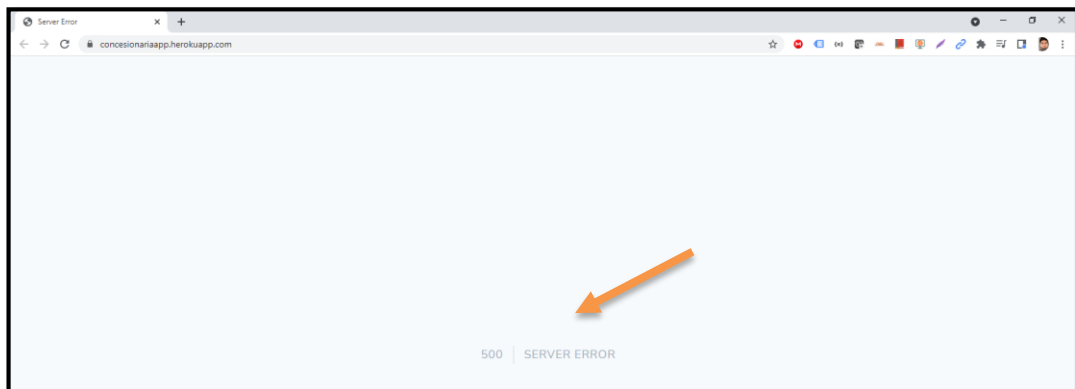
```
PS C:\Users\BYRONTOSH\Desktop\concesionaria> git init
Initialized empty Git repository in C:/Users/BYRONTOSH/Desktop/concesionaria/.git/
PS C:\Users\BYRONTOSH\Desktop\concesionaria> heroku git:remote -a concesionariaapp
» Warning: heroku update available from 7.53.0 to 7.59.1.
set git remote heroku to https://git.heroku.com/concesionariaapp.git
PS C:\Users\BYRONTOSH\Desktop\concesionaria> git add .
warning: LF will be replaced by CRLF in .editorconfig.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in .env.example.
The file will have its original line endings in your working directory
```

```
PS C:\Users\BYRONTOSH\Desktop\concesionaria> git commit -m "Aplicacion_concesionaria"
[master (root-commit) 921dacb] Aplicacion_concesionaria
97 files changed, 11449 insertions(+)
create mode 100644 .editorconfig
create mode 100644 .env.example
create mode 100644 .gitattributes
create mode 100644 .gitignore
```

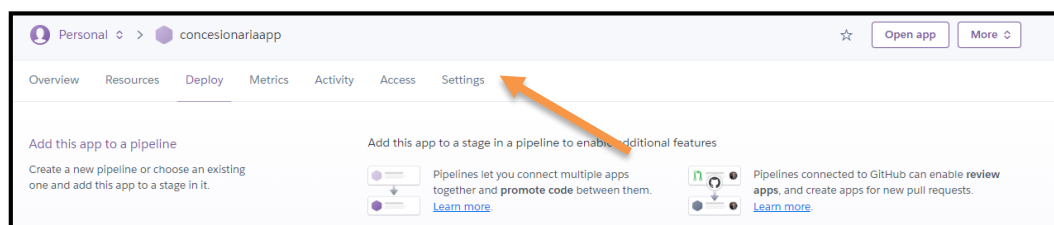
```
PS C:\Users\BYRONTOSH\Desktop\concesionaria> git push heroku master
Enumerating objects: 124, done.
Counting objects: 100% (124/124), done.
Delta compression using up to 4 threads
Compressing objects: 100% (105/105), done.
Writing objects: 100% (124/124), 70.86 KiB | 1.51 MiB/s, done.
Total 124 (delta 7), reused 0 (delta 0), pack-reused 0
```

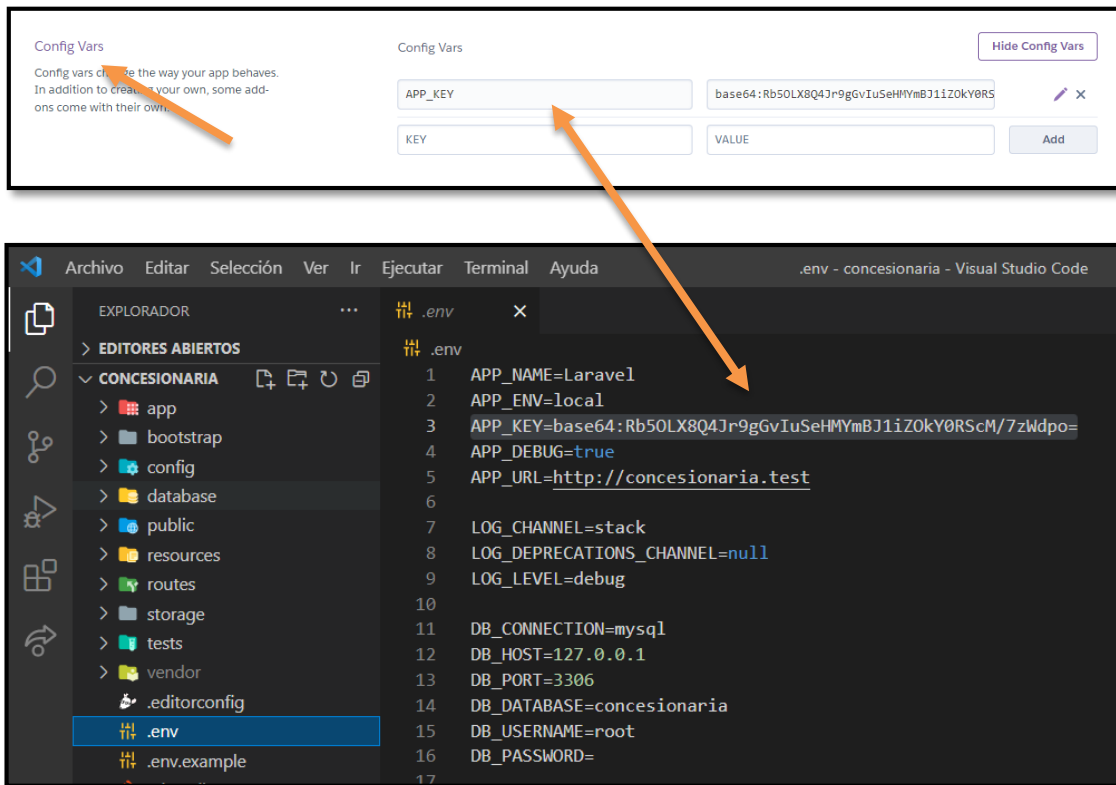
```
remote: - Installing laravel/tinker (v2.6.2): Extracting archive
remote: Generating optimized autoload files
remote: > Illuminate\Foundation\ComposerScripts::postAutoloadDump
remote: > @php artisan package:discover --ansi
remote: Discovered Package: fruitcake/laravel-cors
remote: Discovered Package: laravel/sanctum
remote: Discovered Package: laravel/tinker
remote: Discovered Package: nesbot/carbon
remote: Package manifest generated successfully.
remote: 50 packages you are using are looking for funding.
remote: Use the `composer fund` command to find out more!
remote: -----> Preparing runtime environment...
remote: -----> Checking for additional extensions to install...
remote: -----> Discovering process types
remote: Procfile declares types -> web
remote: -----> Compressing...
remote: Done: 18.7M
remote: -----> Launching...
remote: Released v3
remote: https://concesionariaapp.herokuapp.com/ deployed to Heroku
remote: Verifying deploy... done.
To https://git.heroku.com/concesionariaapp.git
 * [new branch]      master -> master
PS C:\Users\BYRONTOSH\Desktop\concesionaria> heroku open
» Warning: heroku update available from 7.53.0 to 7.59.1.
PS C:\Users\BYRONTOSH\Desktop\concesionaria> 
```

## Verificar

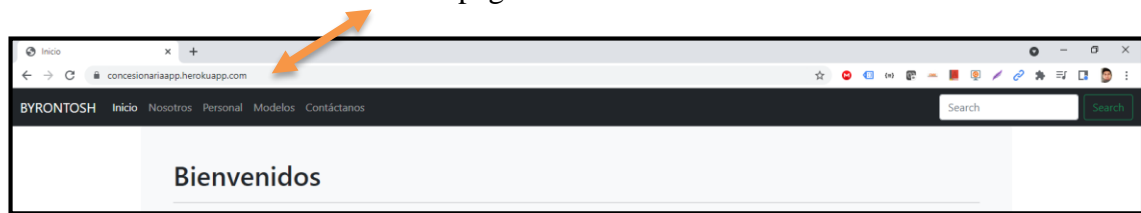


Ahora en la pestaña settings de heroku se procede a establecer la variable de entorno, esta misma variable es la misma del archivo .env del proyecto.





Verificar nuevamente con un F5 a la página.



<https://concesionariaapp.herokuapp.com/>

3. Recuerda que todas las peticiones a nivel de URL deben ser resueltas por un protocolo HTTPS



#### Referencias bibliográficas

- <https://stackoverflow.com/questions/31467871/setting-up-https-redirects-on-heroku-laravel-instance>
- [https://shouts.dev/articles/laravel-dd-vs-dump-vs-var\\_dump-vs-printr-with-example](https://shouts.dev/articles/laravel-dd-vs-dump-vs-var_dump-vs-printr-with-example)