**Manual de instalación del proyecto**

**Modulo Back-end v1**

**Juan Pablo Acevedo Torres**

**Modulo Front-end**

**Juan Ulloa**

**MODULO BACKEND CMS**

**Juan Pablo Acevedo Torres**

**Cod 3110977**

**Universidad Santo Tomas**

**Tunja**

Tabla de contenido

[Requerimientos y guías de trabajo 5](#_Toc5277341)

[Crear proyecto 5](#_Toc5277342)

[Crear proyecto desde cero 5](#_Toc5277343)

[Clonar repositorio de proyecto 6](#_Toc5277344)

[CONFIGURACION INICIAL 6](#_Toc5277345)

[Crear base de datos 7](#_Toc5277346)

[Modelo conceptual base de datos 7](#_Toc5277347)

[Creación de las tablas 8](#_Toc5277348)

[configurar driver de conexión de postgresql “pgsql” 9](#_Toc5277349)

[Crear proyecto desde CMD. 9](#_Toc5277350)

[Proyecto base 9](#_Toc5277351)

[Cambiar propiedades en el archivo composer.json 10](#_Toc5277352)

[Cambiar el nombre del proyecto app: 11](#_Toc5277353)

[Colocar el proyecto en modo desarrollador 11](#_Toc5277354)

[Configurar timezone e idioma por defecto 11](#_Toc5277355)

[Configuración base de datos por defecto 12](#_Toc5277356)

[Creacion de los modelos de la base de datos en laravel 13](#_Toc5277357)

[Modulo de auntenticacion 14](#_Toc5277358)

[Crear funciones de registro 15](#_Toc5277359)

[Control de Usuario 15](#_Toc5277360)

[Librerías y dependencias necesarias 15](#_Toc5277361)

[Función de registro por defecto de tipo usuario. 15](#_Toc5277362)

[Rutas de las funciones get y post 16](#_Toc5277363)

[Pruebas de conexión intranet 17](#_Toc5277364)

[Manejo de errores 19](#_Toc5277365)

[Runtime Exception: 19](#_Toc5277366)

[Error en la migración 20](#_Toc5277367)

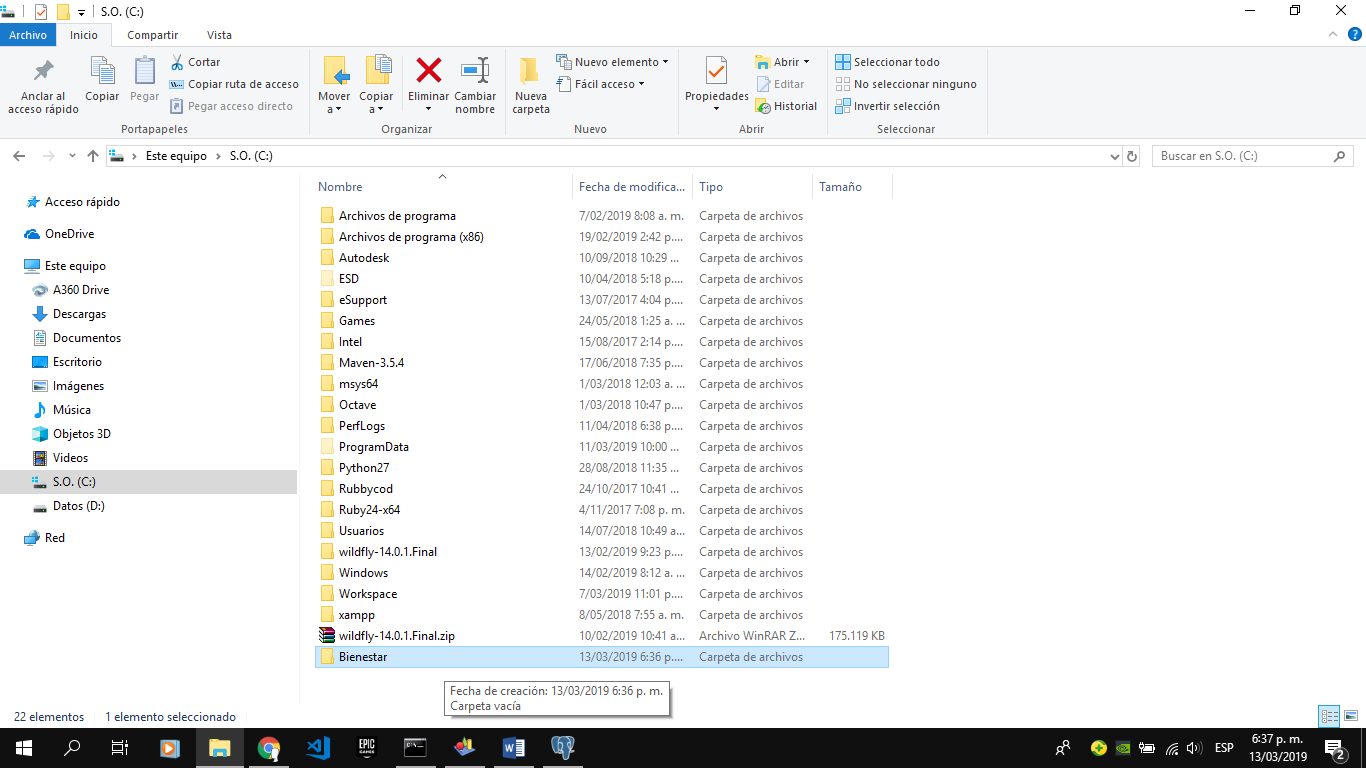
# Requerimientos y guías de trabajo

|  |  |
| --- | --- |
| **Guia** | **descripción** |
| Diseño Sistema web PHP con laravel y Mysql | <https://www.youtube.com/playlist?list=PLZPrWDz1MolrxS1uw-u7PrnK66DCFmhDR> |
| Introducción a Laravel 5 - Primeros pasos con este framework | <https://www.udemy.com/introduccion-a-laravel-5-primeros-pasos-framework-php/> |

|  |  |  |
| --- | --- | --- |
| **Programa o librerías** | **versión** | **Link** |
| Laravel | 5.4 | <https://laravel.com/docs/5.8/installation> |
| PHP | 7.1 | https://windows.php.net/downloads/releases/php-7.1.27-src.zip |
| Xampp | 7.1.27 | <https://www.apachefriends.org/es/download.html> |
| Editor de código visual code o sublime text |  | <https://code.visualstudio.com/>  <https://www.sublimetext.com/> |
| Postgres version estable | 9.6 | <https://www.openscg.com/bigsql/postgresql/installers.jsp/> |
| github |  | <https://github.com/> |
| idiomas | No es obligatorio | <https://github.com/caouecs/Laravel-lang> |

# Crear proyecto

## Crear proyecto desde cero



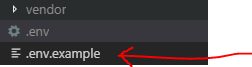
## Clonar repositorio de proyecto

La clonación de este proyecto, solo tiene permisos de copia y no de edición a menos que se le solicite las credenciales de acceso al desarrolador.

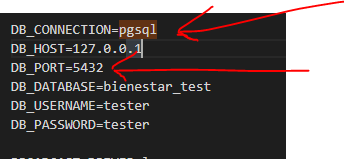
<https://github.com/juanpabloacevedo/bienestar.git>

# CONFIGURACION INICIAL

crear copia del archivo **.env.example** y cambiar nombre a **.env** , este es el archivo de configuración del proyecto en nuestro equipo, luego en la consola de comandos ejecutar composer update para instalar el proyecto y luego usar el comando **php artisan key:**generate para crear la identificación única de nuestro proyecto en nuestro equipo, si no generamos la llave “key ”nos generara un error a la hora de compilar el proyecto *’ver control de fallos’*



debe aparecer lo siguiente en el archivo **.env** configurado para postgresql ”**pgsql**”

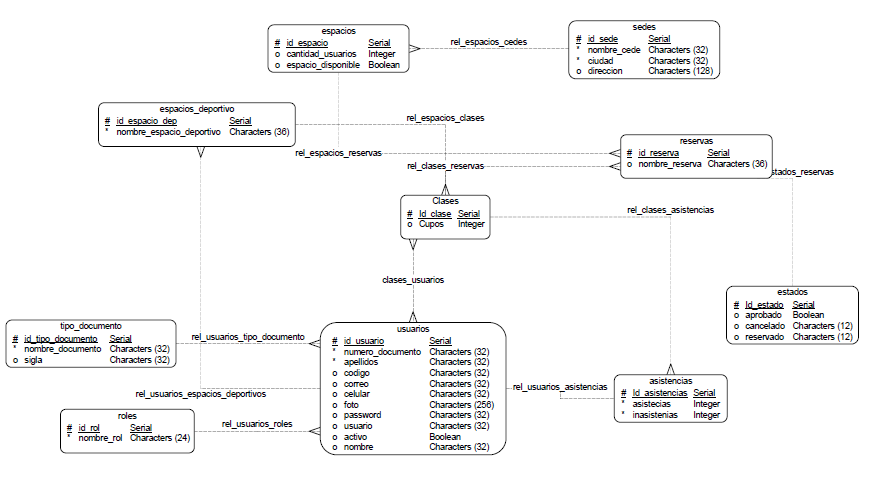


Si la base de datos no existe debe crearse, más adelante se muestra cómo.

# Crear base de datos

Nombre y configuración base de datos

|  |  |
| --- | --- |
| Nombre base de datos | bienestar\_test |
| owner | tester |
| password | tester |
| Port | 5432 |

Modelo conceptual base de datos 

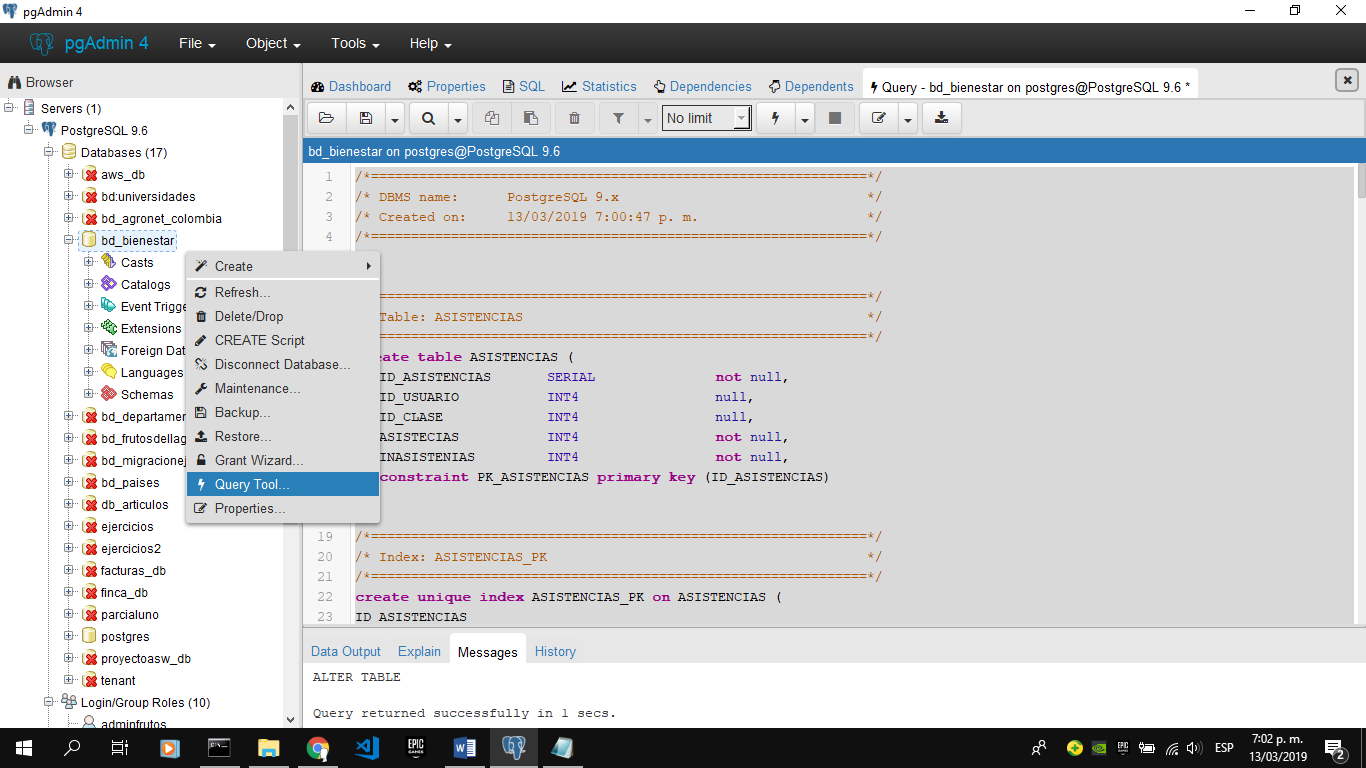
Base de datos en postgresql 9.6

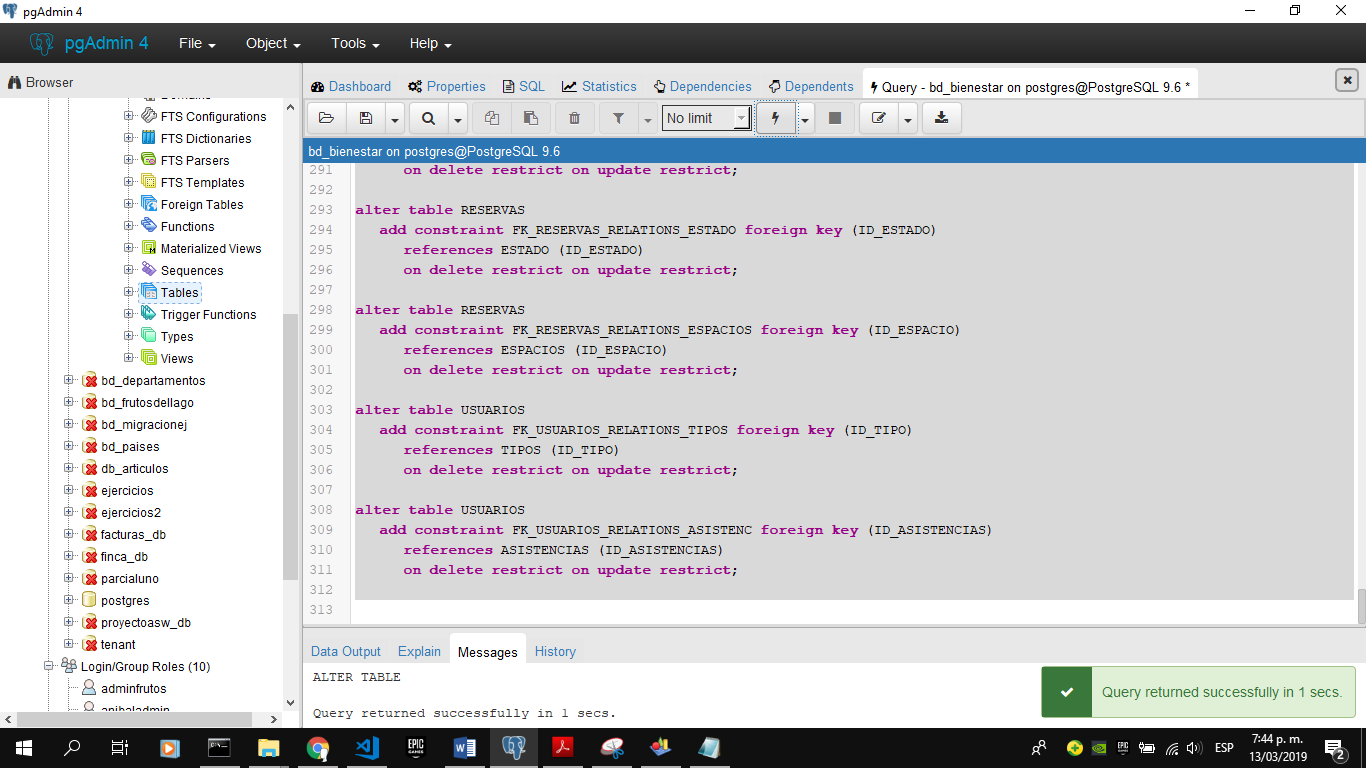


## Creación de las tablas

Para la creación de las tablas tenemos dos opciones

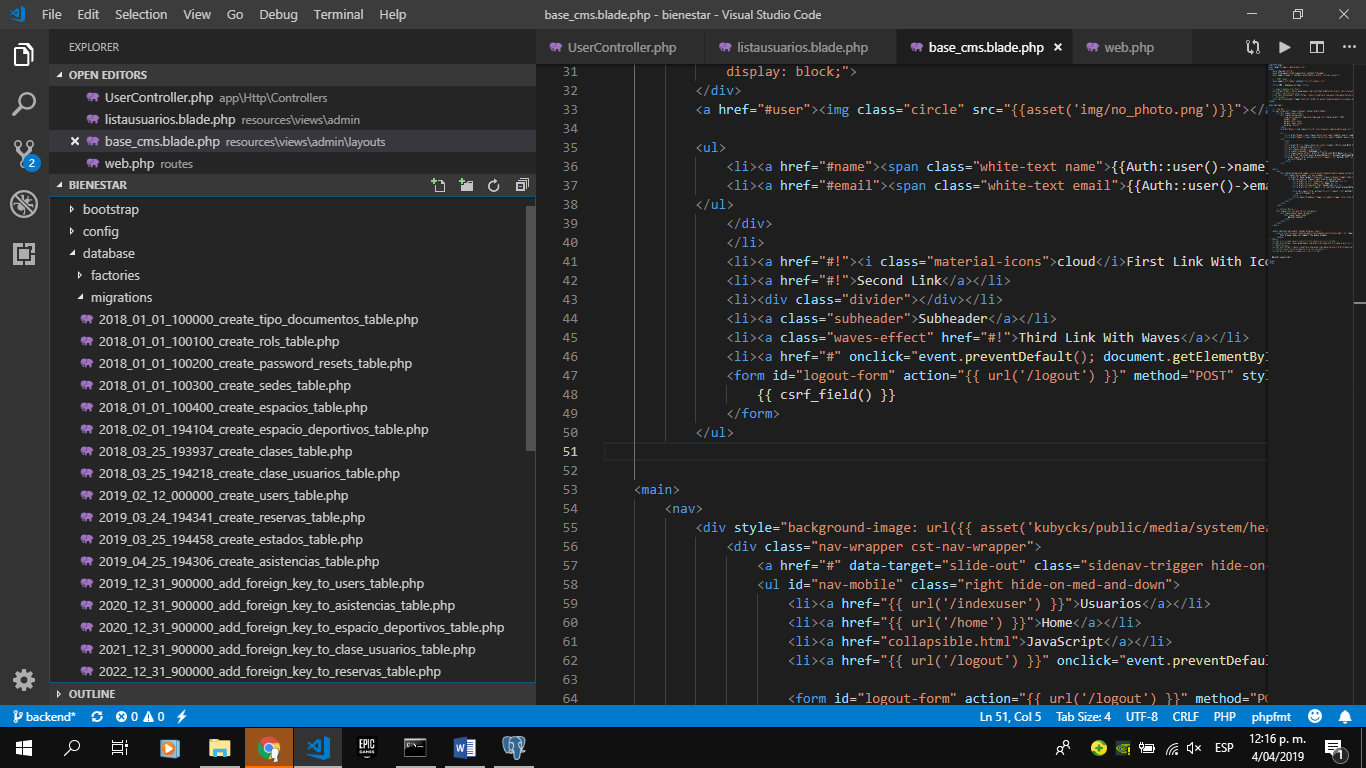
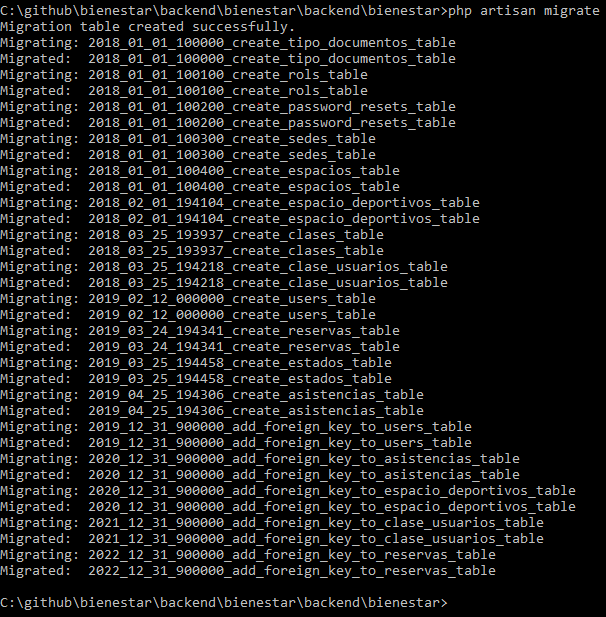
1. abrir el query tool de la base de datos bienestar\_bd y crear las tablas usando el archivo DBMS adjunto en el repositorio





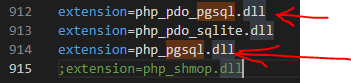
1. usar las migraciones que vienen en el archivo de configuración de base de datos, pero esto solo se puede hacer si el proyecto laravel ya existe.

para esto usamos el comando **php artisan migrate**

## configurar driver de conexión de postgresql “pgsql”

por defecto php ya trae un dll de conexión, pero hay que activarlo. En **xampp**, dentro de la carpeta php buscar archivo php.ini y descomentariar “extension=php\_pdo\_pgsql.dll” para usar el driver de postgresql posteriormente.

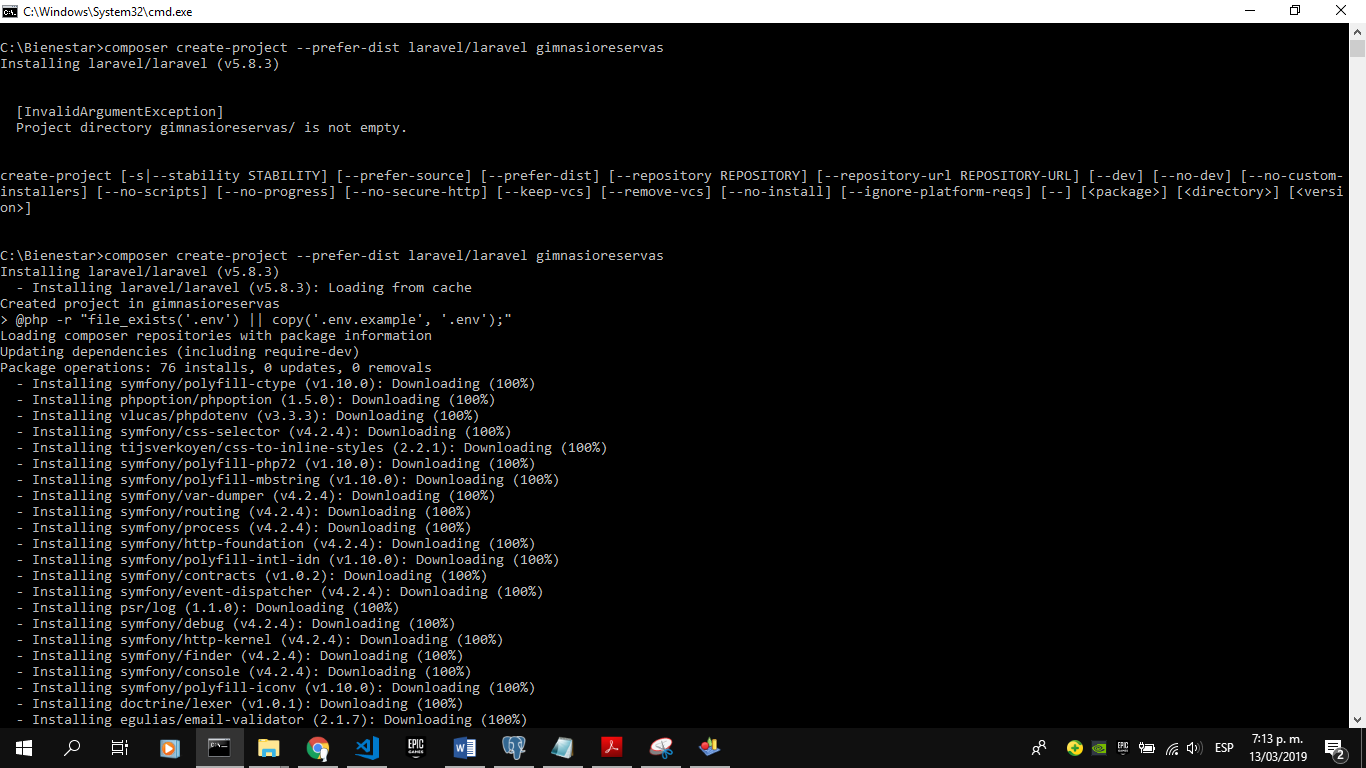


# Crear proyecto desde CMD.

Proyecto base.

Abrir la consola de comandos ,abrir la carpeta Bienestar, luego backend y dentro de esta digitar:

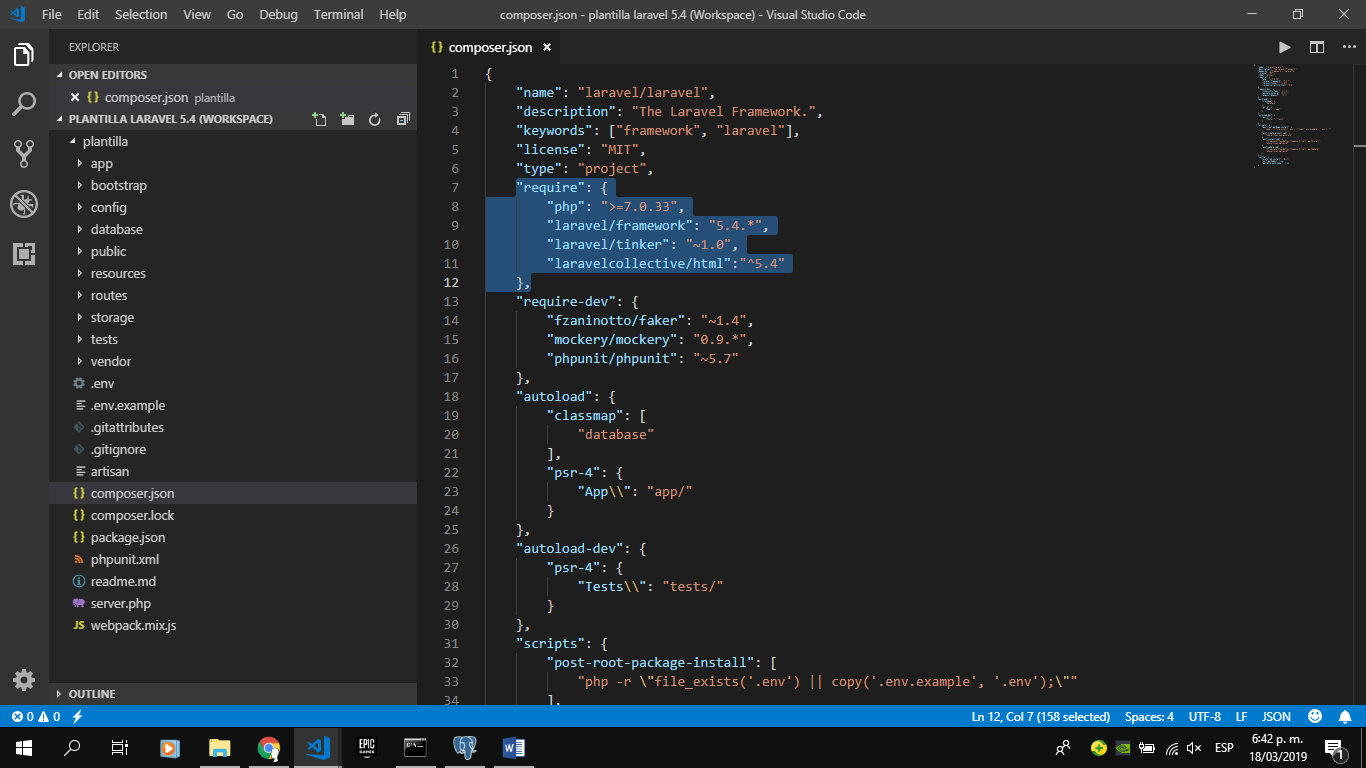
**composer create-project --prefer-dist laravel/laravel bienestar**



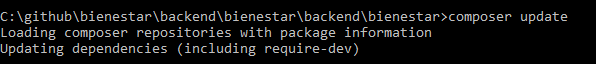
La instalacion puede tardar 10 minutos aproximadamente, dependiendo de la cantidad de librerias a descargar y del equipo

## Cambiar propiedades en el archivo composer.json

Una vez instalado el proyecto base, procedemos a configurarlo y descargaremos las libererias que nesesitemos.

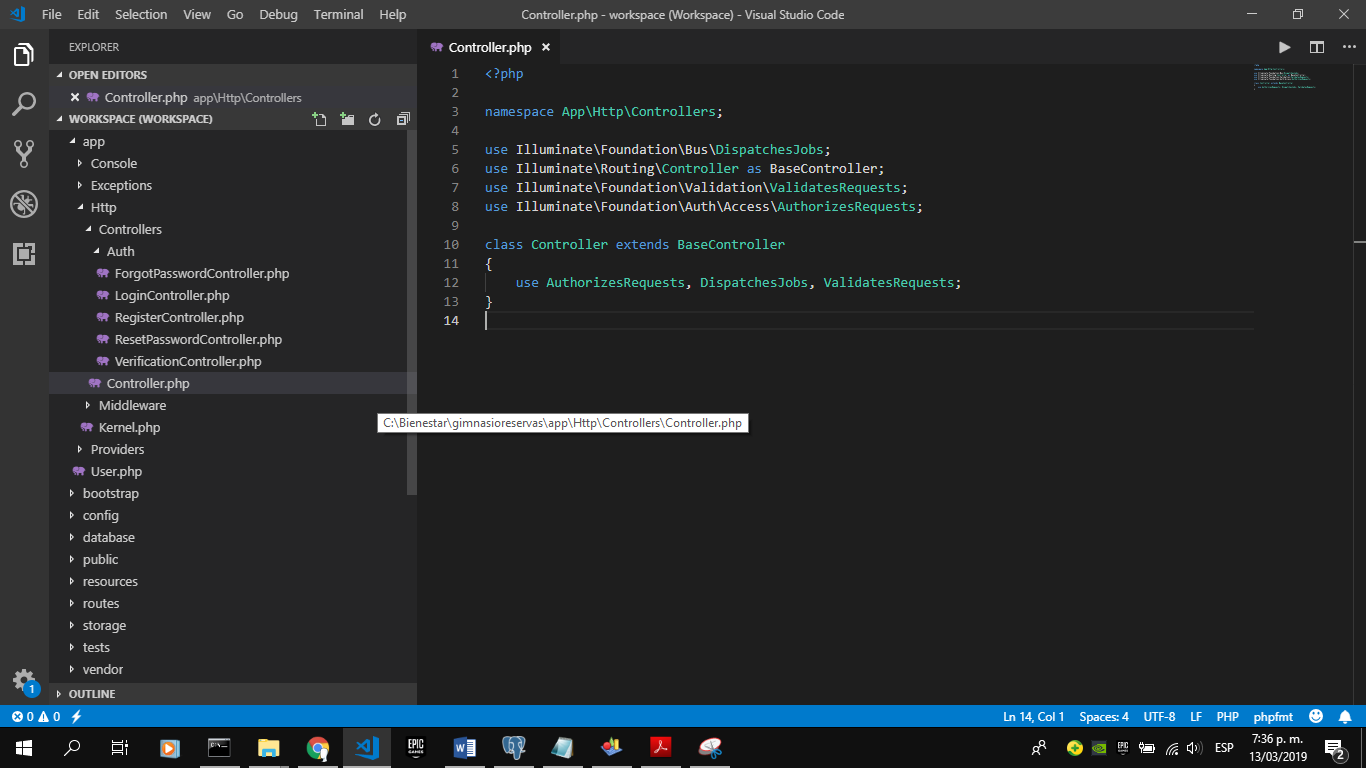


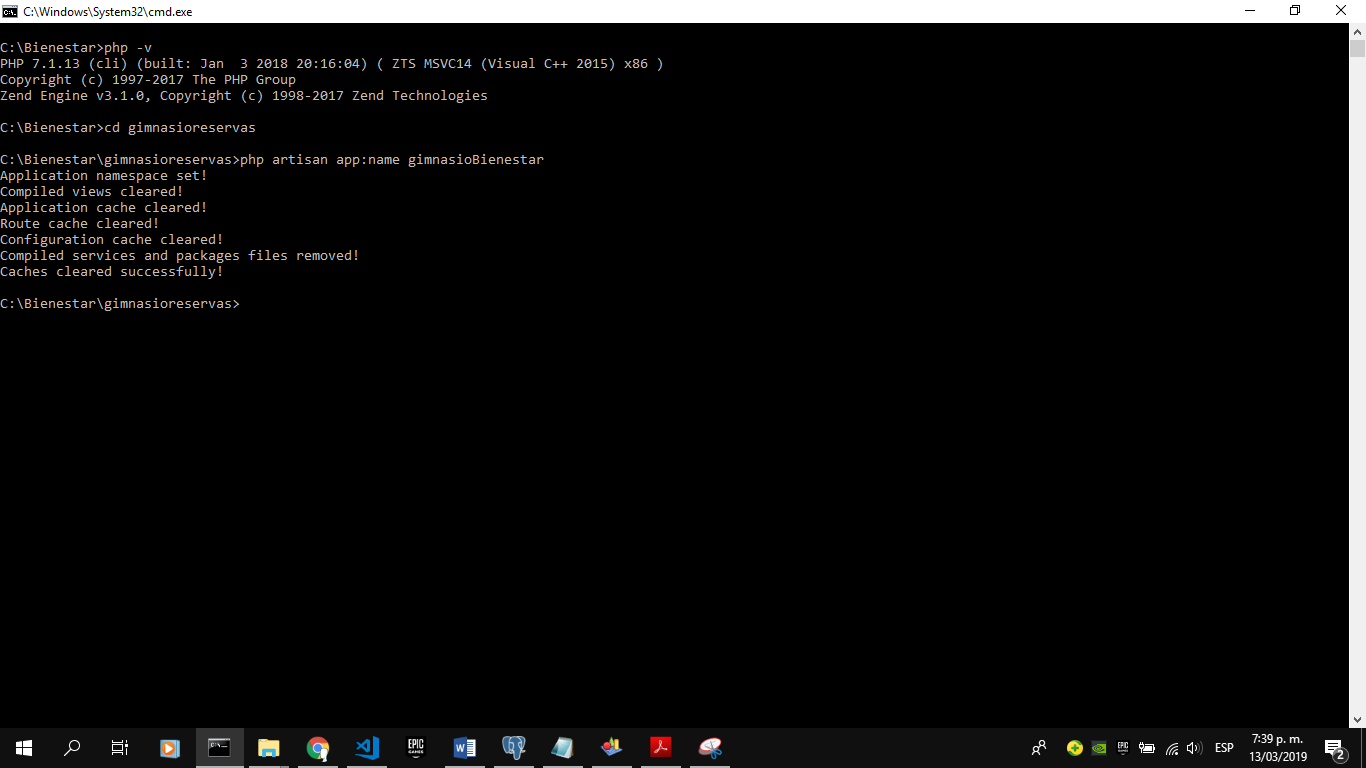
En consola digitar el comando “**composer update**” dentro de la ubicación del proyecto



## Cambiar el nombre del proyecto app:

Esta acción no es necesaria, ya que puede generar errores de enrutamiento por la dependencia del nombre App





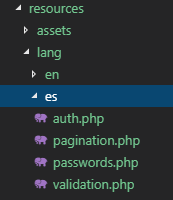
## Colocar el proyecto en modo desarrollador

### Configurar timezone e idioma por defecto

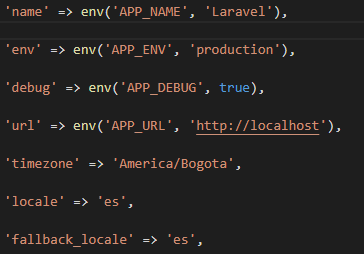
En la carpeta **config** nos dirigimos al archivo app.php, en este archivo configuraremos el timezone del proyecto para que coincida con la configuración de tiempo de nuestra zona horaria, también el idioma por defecto, aunque para esto hay que buscar un repositorio de idiomas y colocaremos el proyecto en modo de pruebas

|  |  |
| --- | --- |
| Repositorio de idiomas usado | <https://github.com/caouecs/Laravel-lang> |
| Zonas horarias permitidas en php | <https://www.php.net/manual/es/timezones.america.php> |

En resources, creamos la carpeta es, y colocamos los 4 archivos de configuracion del repositorio para el idioma español

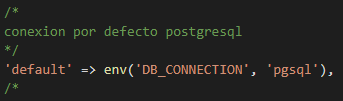


Configuramos el proyecto en modo desarrollador al darle en APP\_DEBUG, true, y digitamos el resto de las configuraciones de la app



### Configuración base de datos por defecto

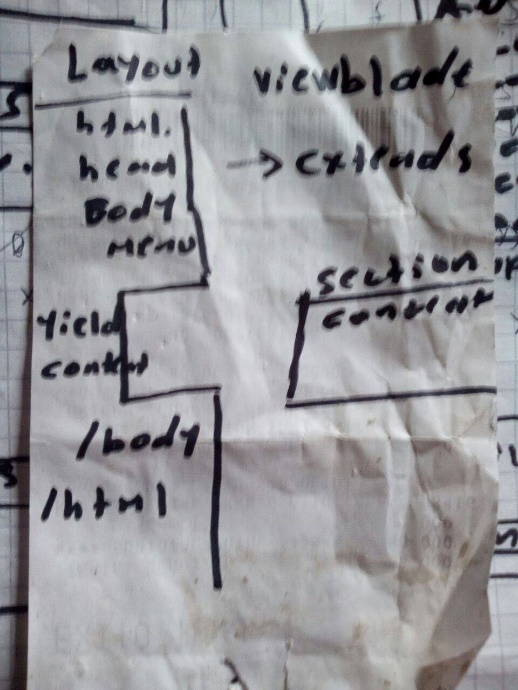
En este caso usaremos PostgreSQL, por lo que nos dirigimos a la carpeta config, luego database.php y cambiamos la configuración por defecto de mysql a pgsql



En el archivo **.env** y en el archivo **.env.example** guardamos la siguiente configuración. Debemos crear primero la base de datos en postgresql

# Flujo de datos

## Versión echa en papel

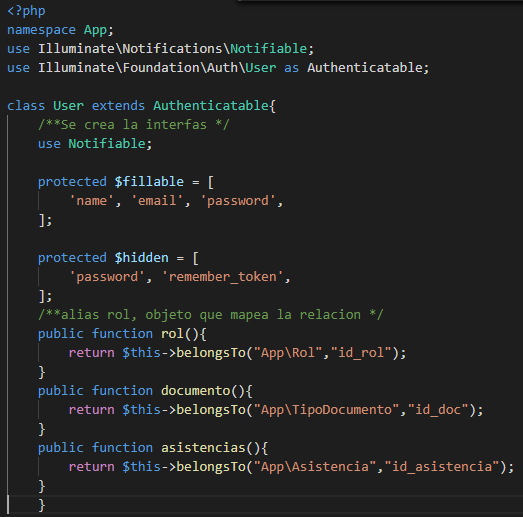
# Creacion de los modelos de la base de datos en laravel

Comando php artisan make:model Nombre\_de\_la\_tabla

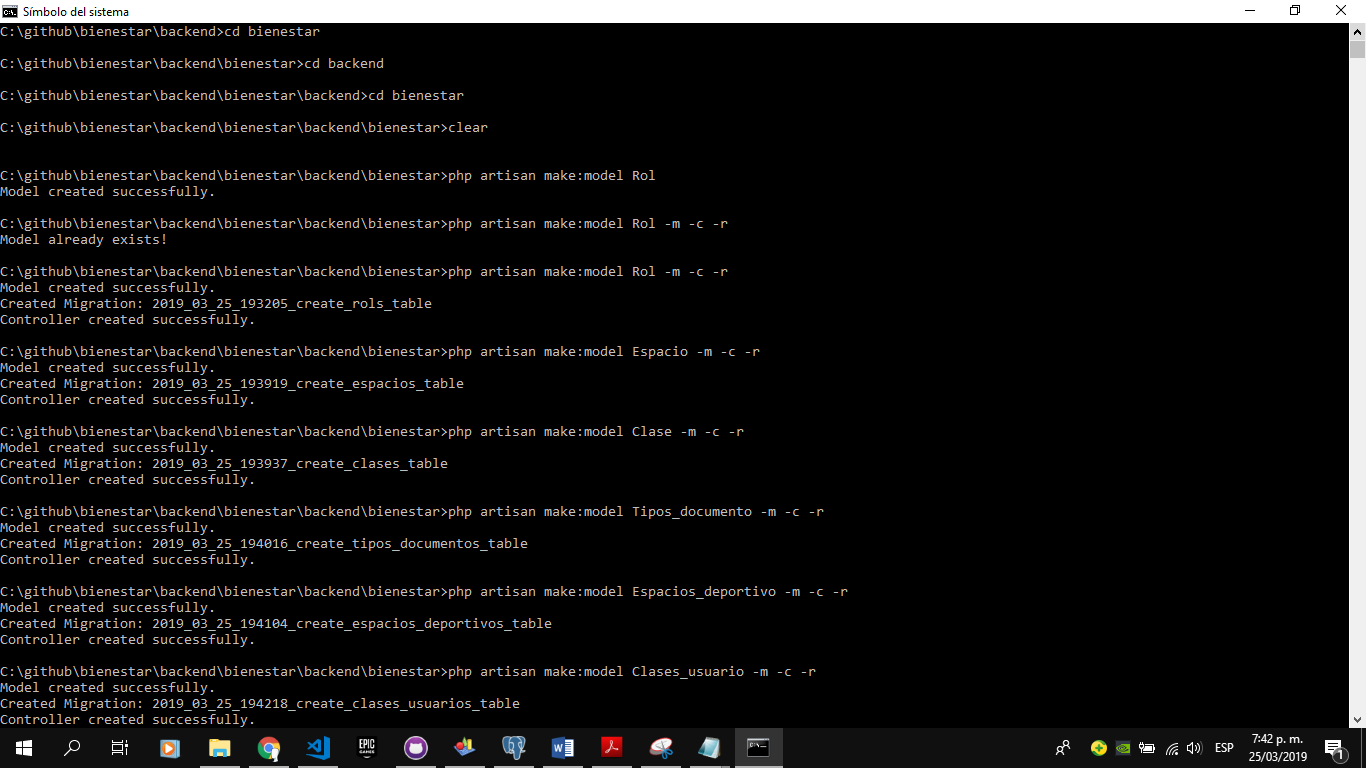
Modelo de la tabla User: con modelo, migracion controlador y resource con el comado php artisan make:model -m -c -r



En el editor de código nos situamos sobre la carpeta app, luego sobre el archivo User.php

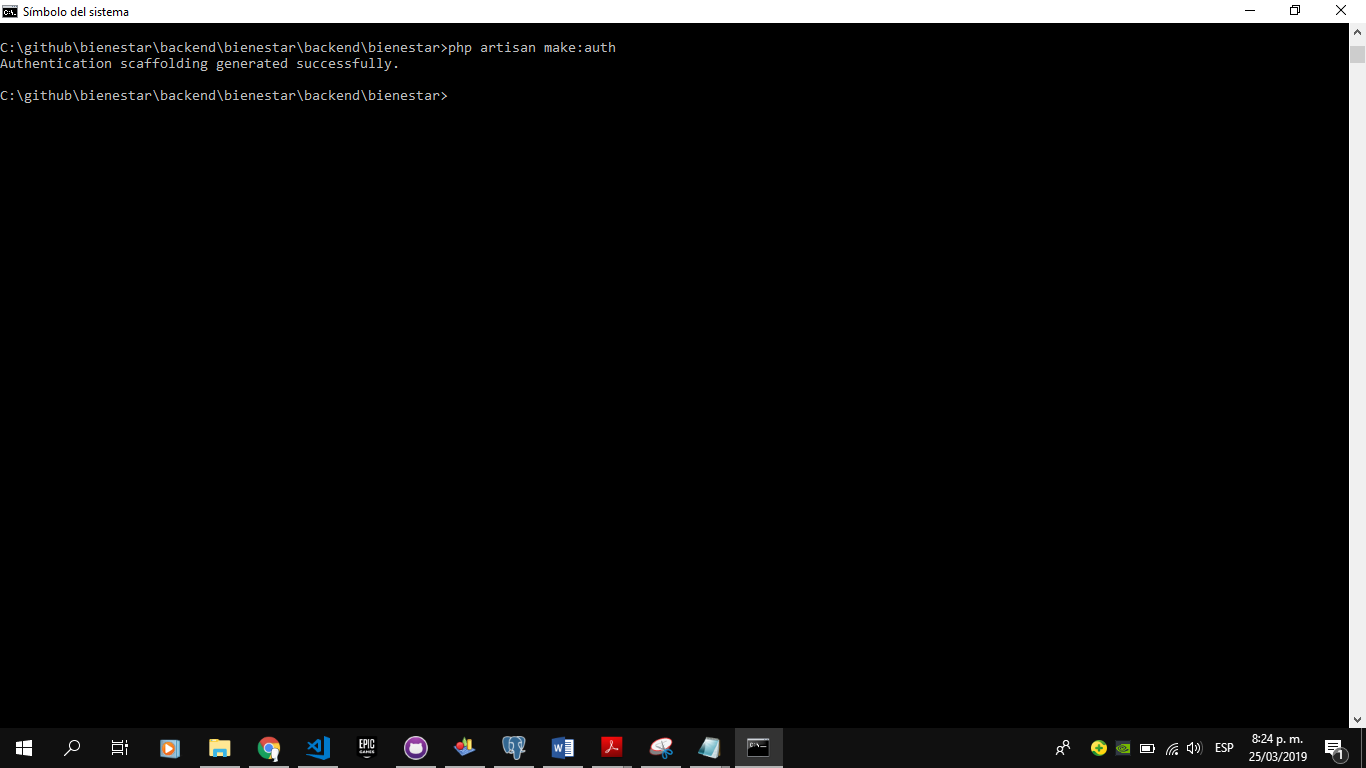


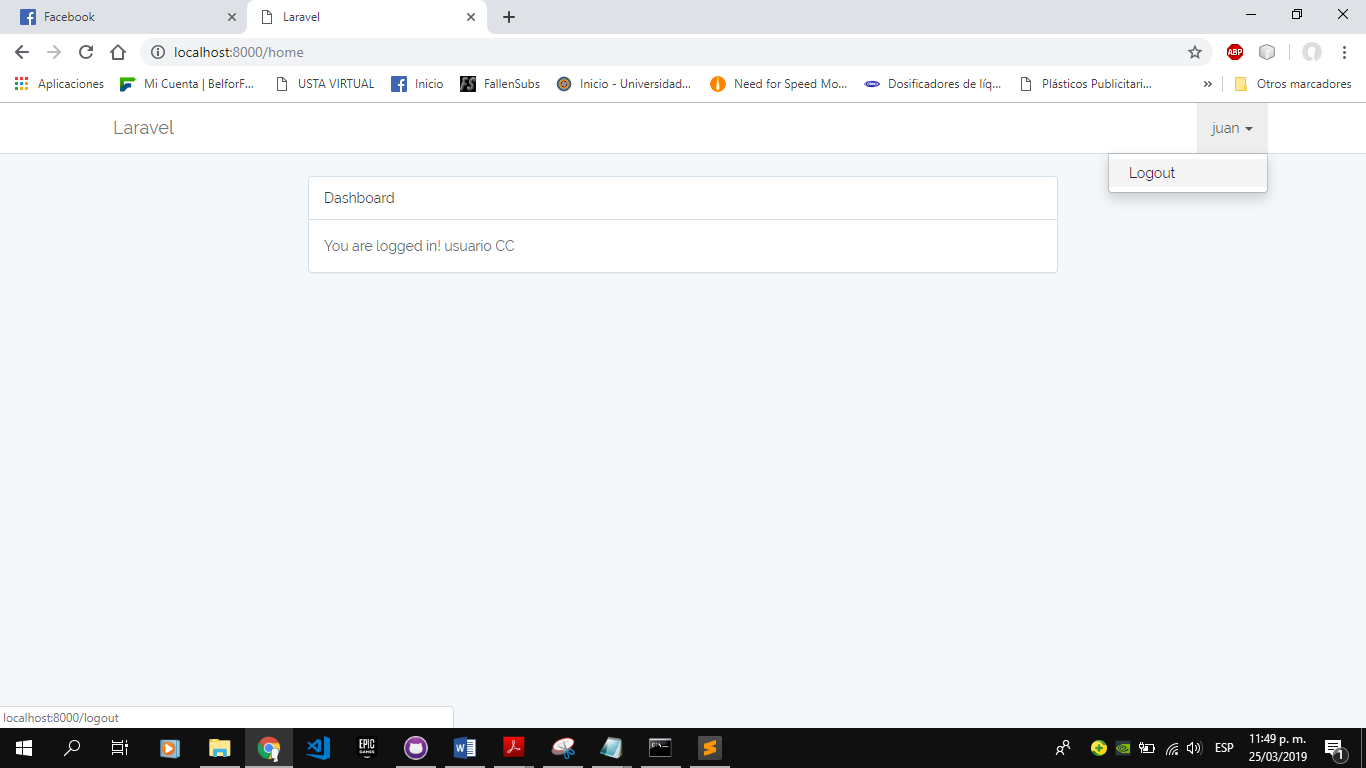
Estos modelos por defecto estarán vacíos al principio, y solo llevarán las relaciones con otras tablas.



## Modulo de auntenticacion

En esta sección crearemos nuestro modulo de autenticación por defecto e trae laravel



Pantalla Home por defecto de usuario registrado y logueado

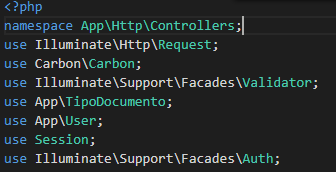
# Crear funciones de registro

En cada controlador crearemos una función de registros propios para los formularios.

## Control de Usuario

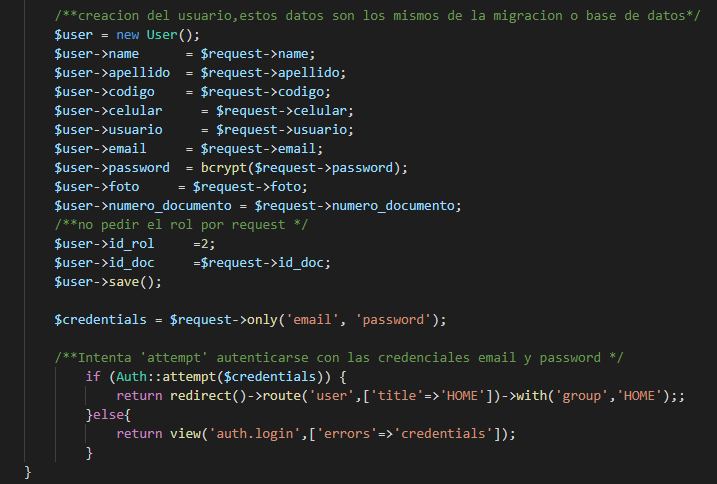
En este ejemplo se mostrará cómo funciona el registro de usuarios

### Librerías y dependencias necesarias

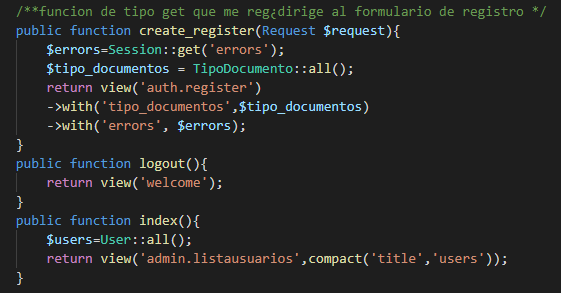


### Función de registro por defecto de tipo usuario.

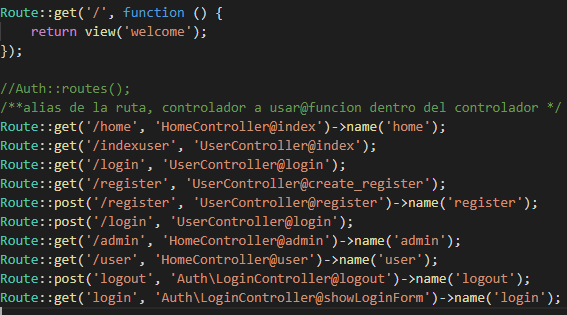
Función de registro “register” de tipo ‘POST’.

Función de registro “créate\_register” de tipo ‘GET.

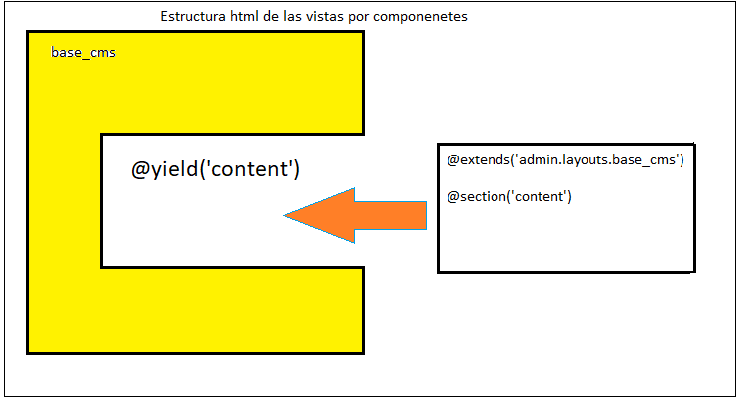


## Rutas de las funciones get y post

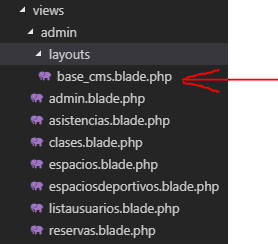
Control de rutas de las funciones de los controladores y las vistas 

# Vistas del módulo administrador

## Estructura



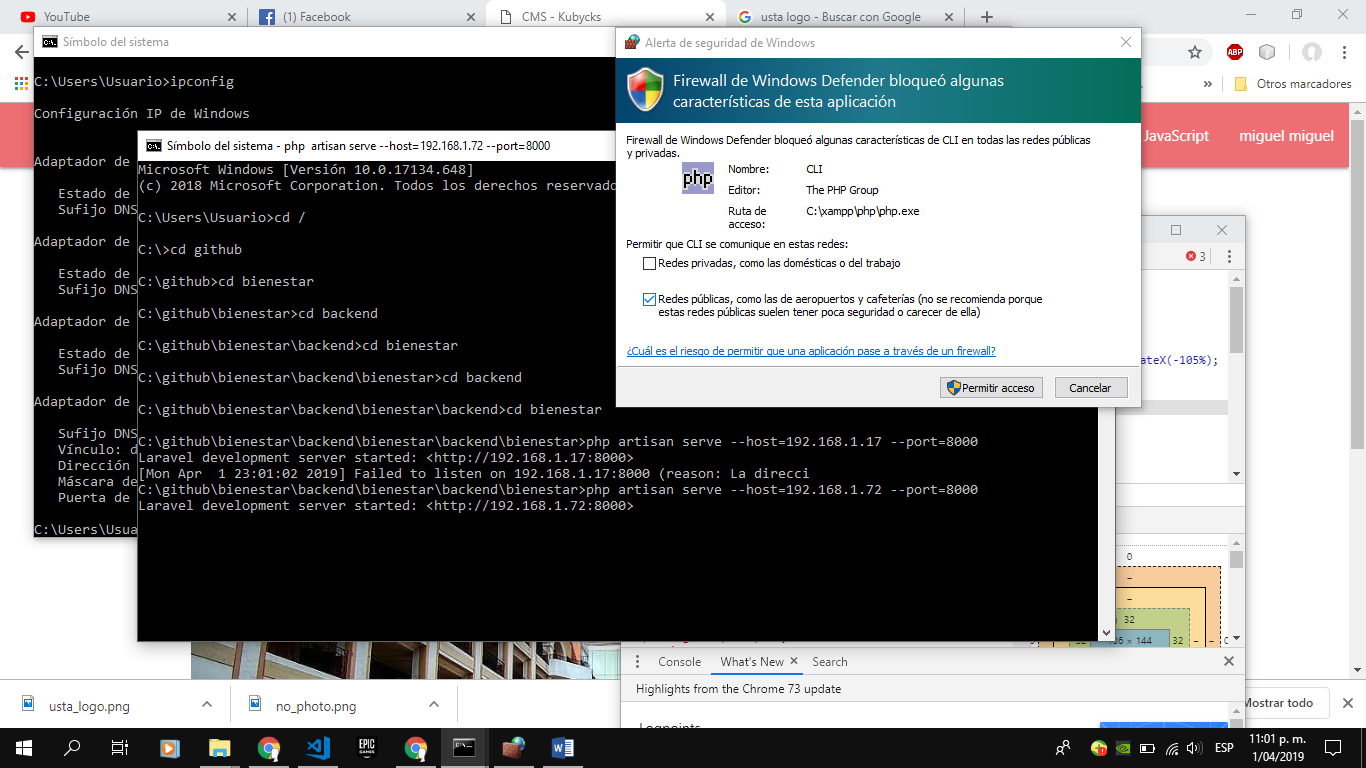
Dentro de la carpeta de las vistas de administrador, crearemos un template genérico, que contendrá un contenido que no cambaiara, como menus, head y footer, y dejaremos un espacio “@yield(’seccion a insertar’)” para insertar el contenido variable de cada vista, como por ejemplo la lista de usuarios, espacios,clases etc.

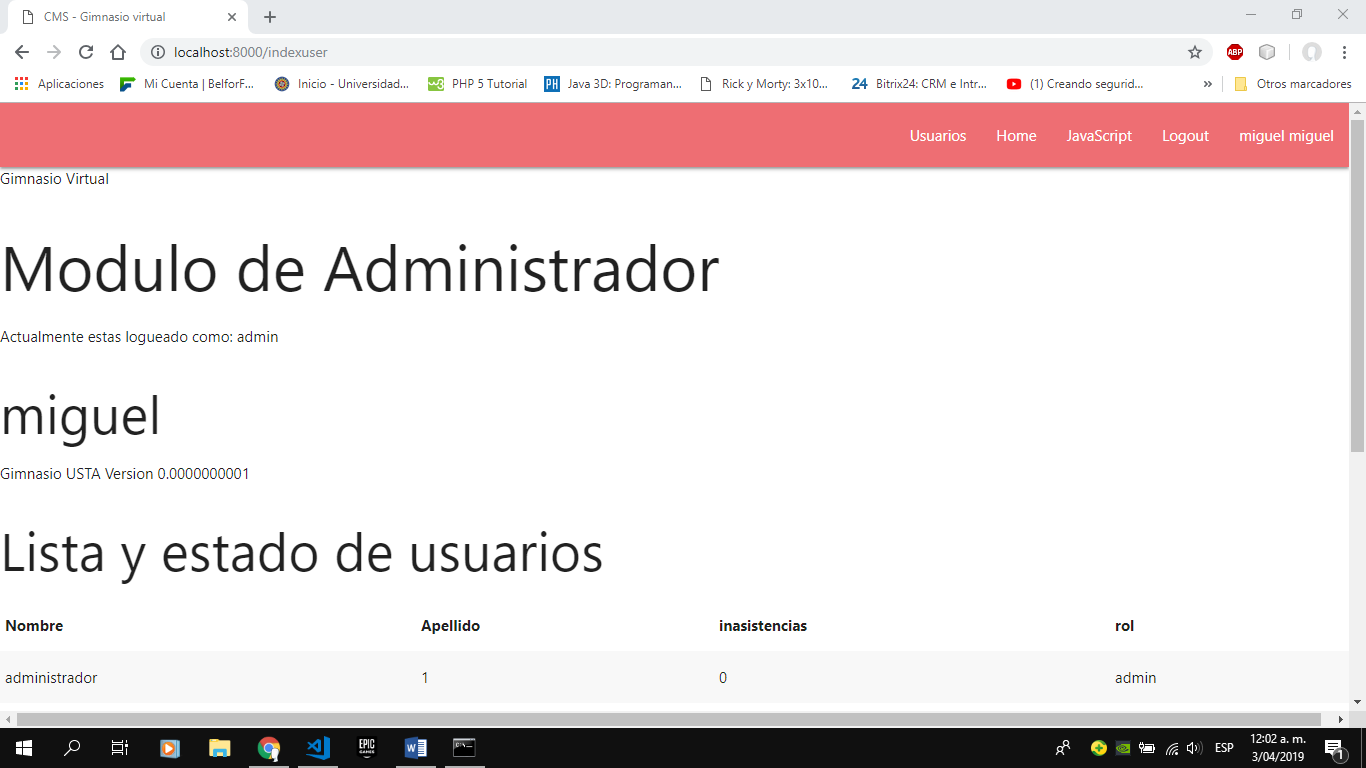


# Pruebas de conexión intranet

Primero buscamos la dirección ip de nuestro equipo en la red, y sobre esta dirección crearemos una intranet de pruebas

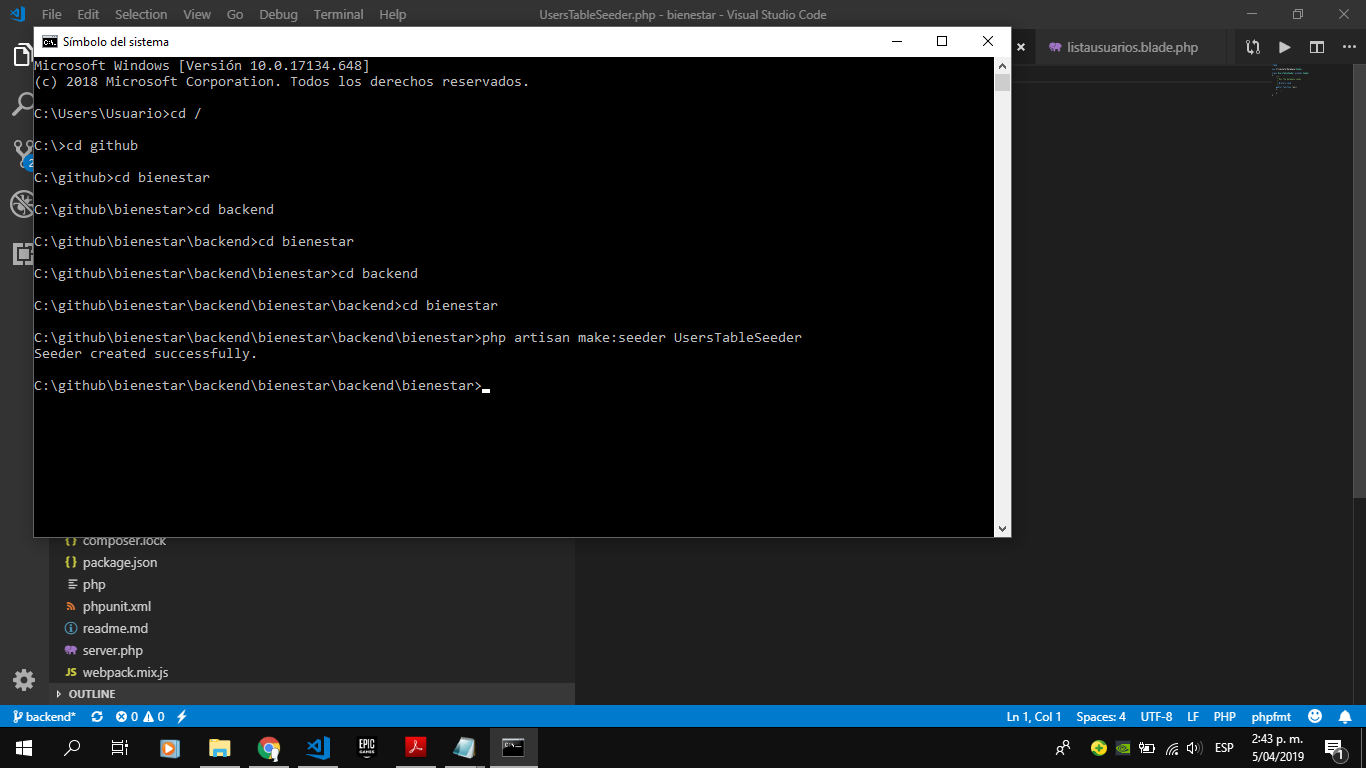




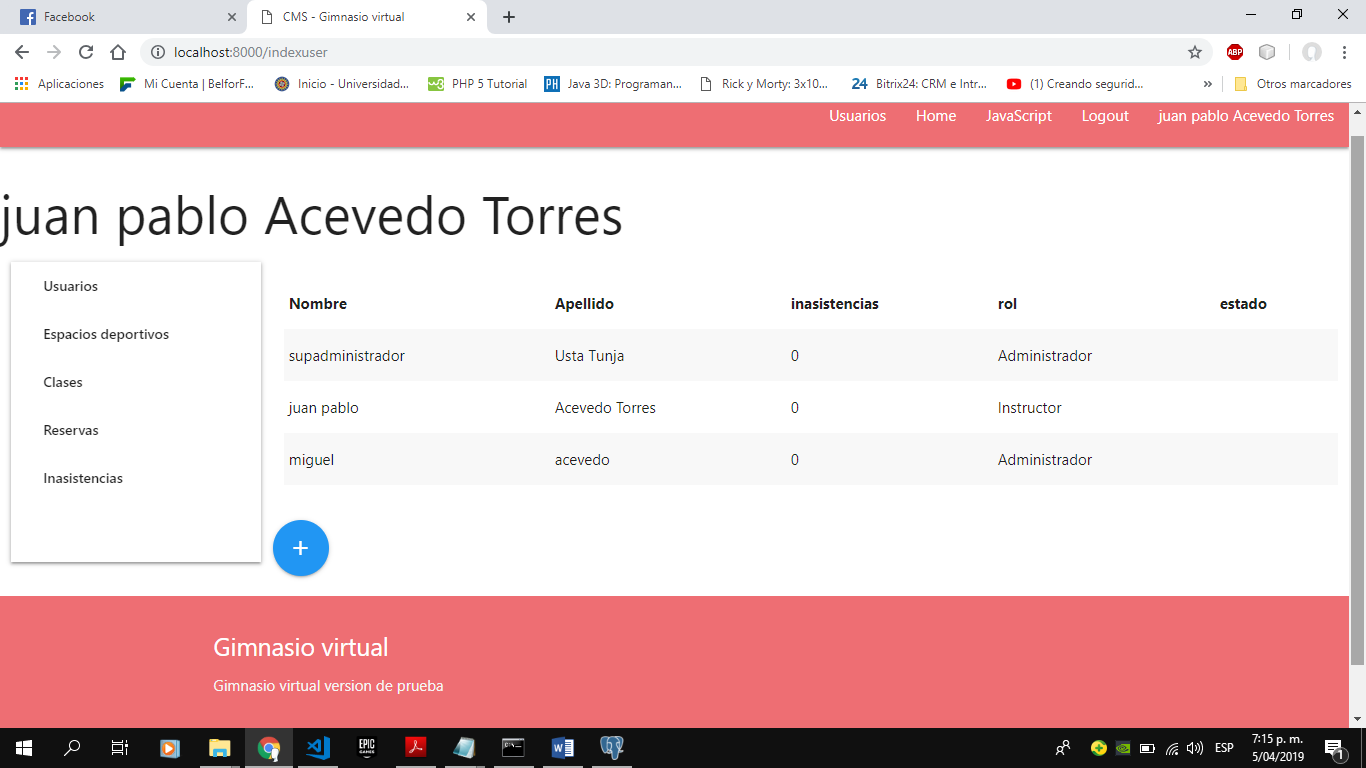


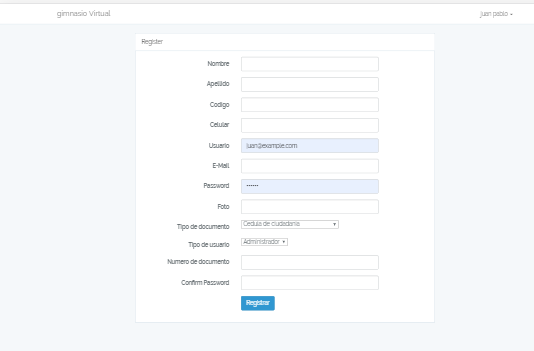
# Datos por defecto para el funcionamiento de la pagina

## Creacion de los datos iniciales con la funcion seeder



# Modulo de usuarios funcional

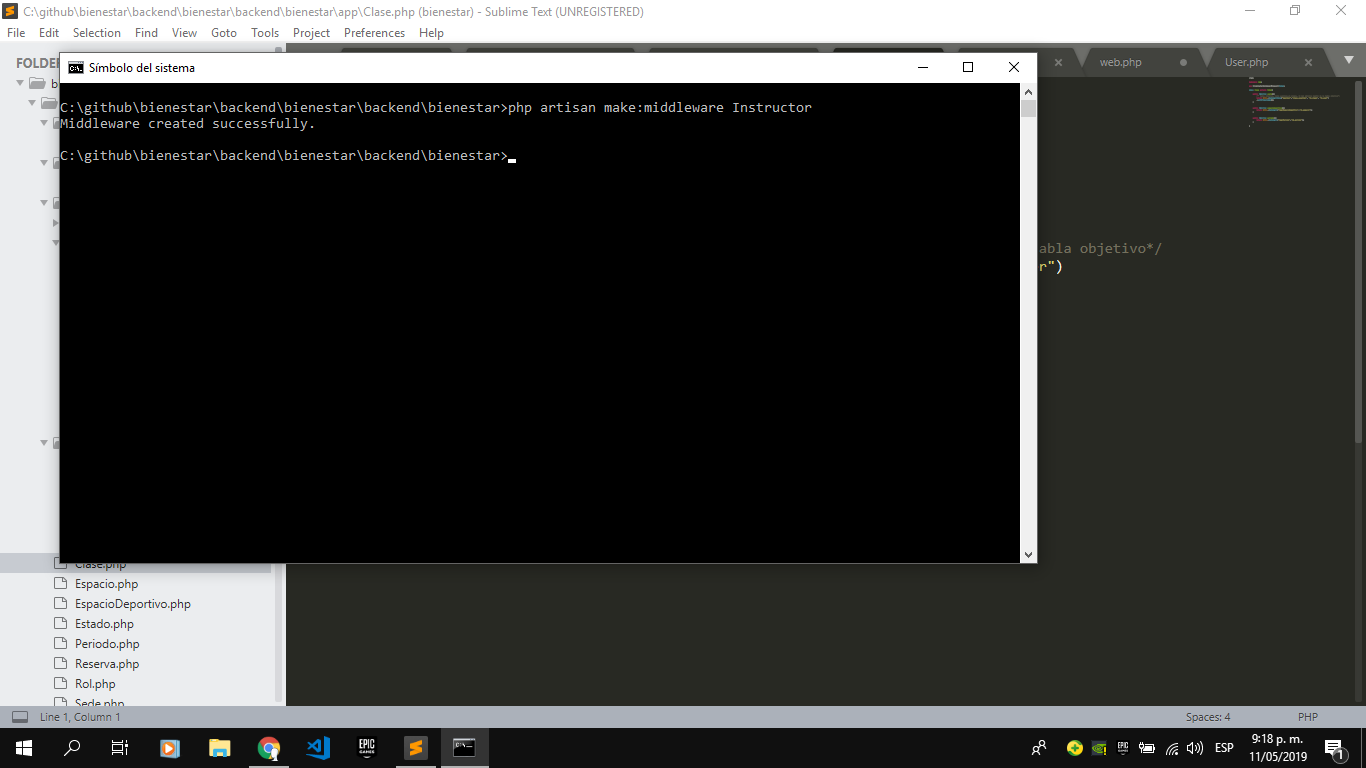


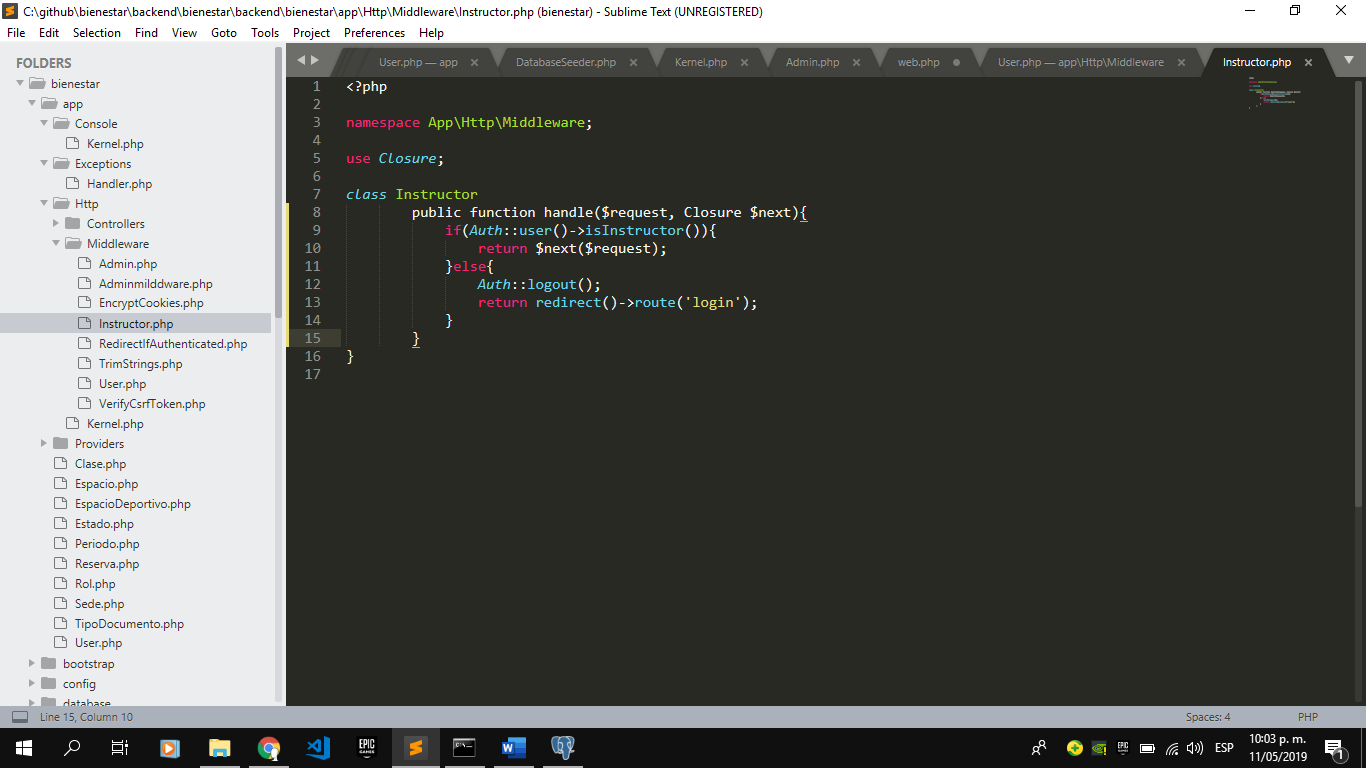


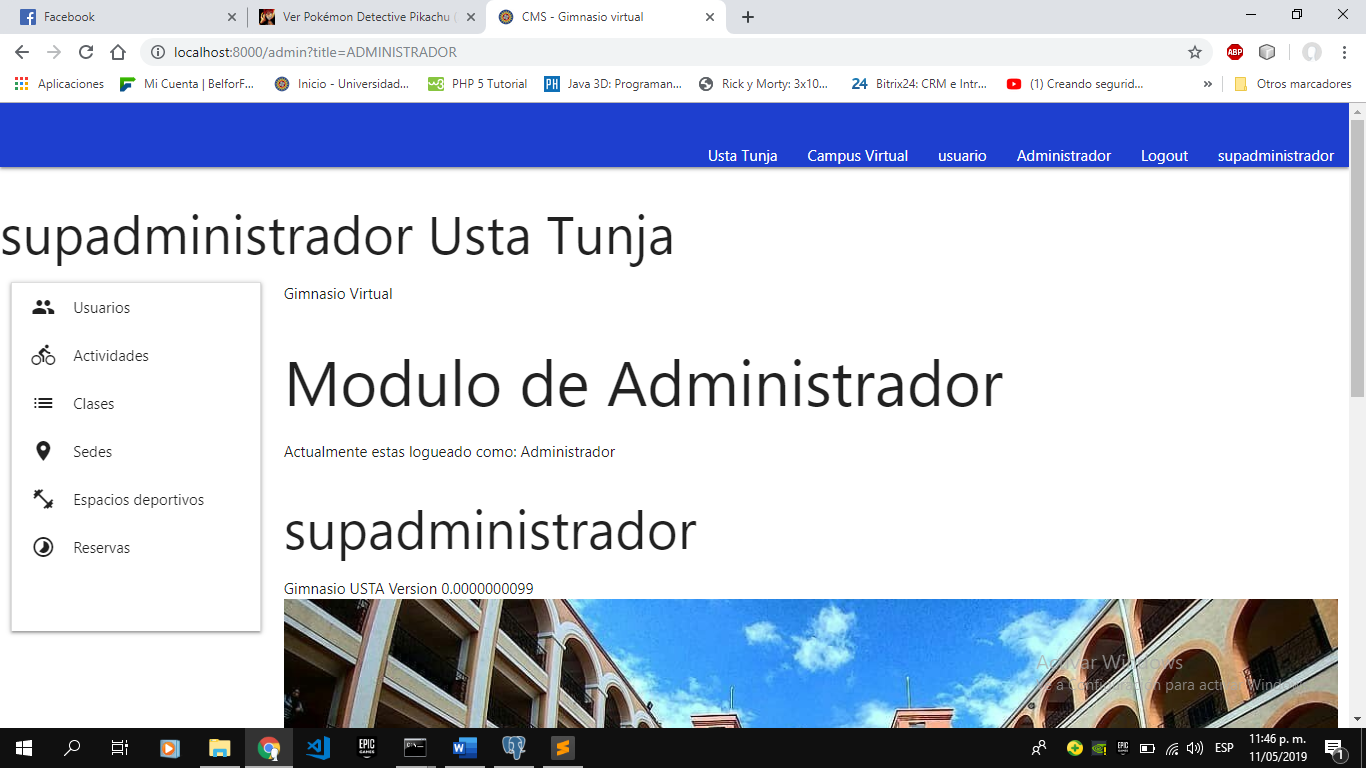
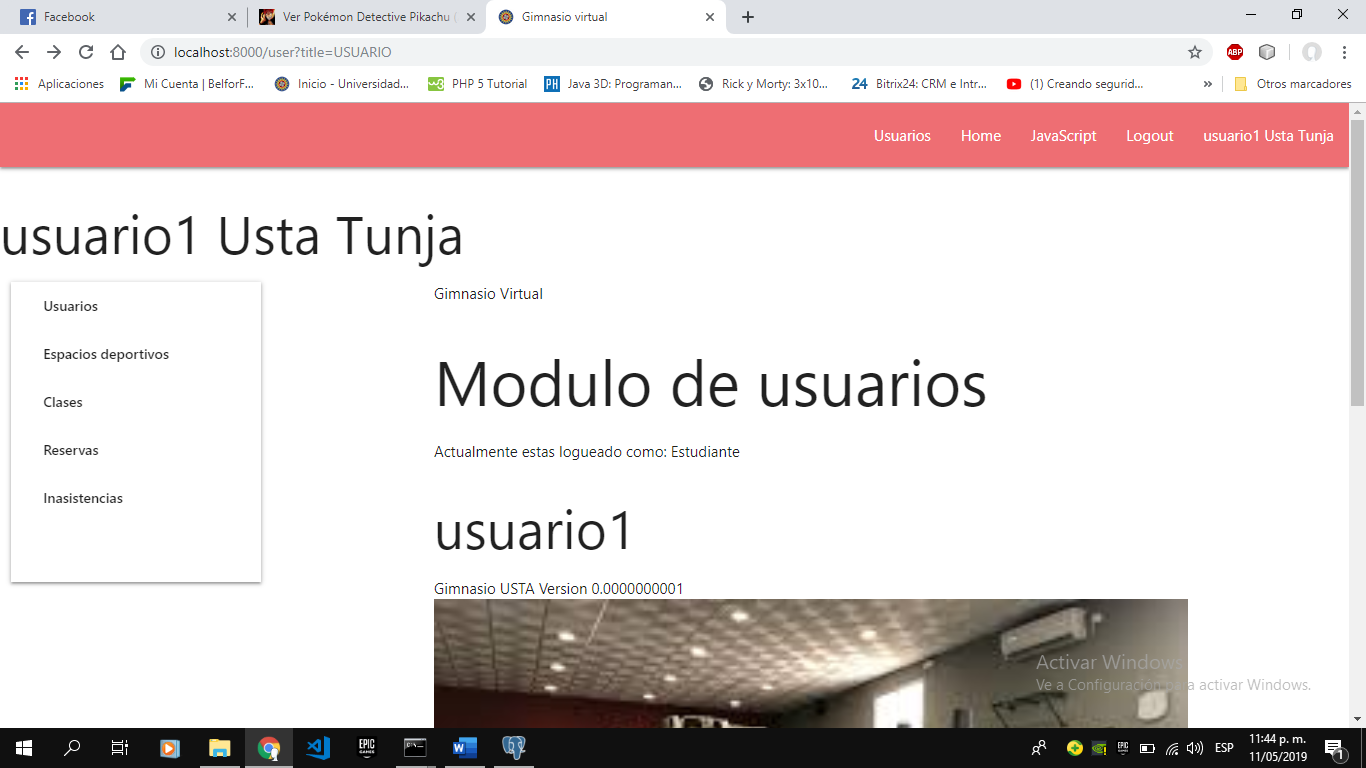
# Filtros de acceso por rol

## Creación de middleware

Creación del modulo de filtro middleware para instructor

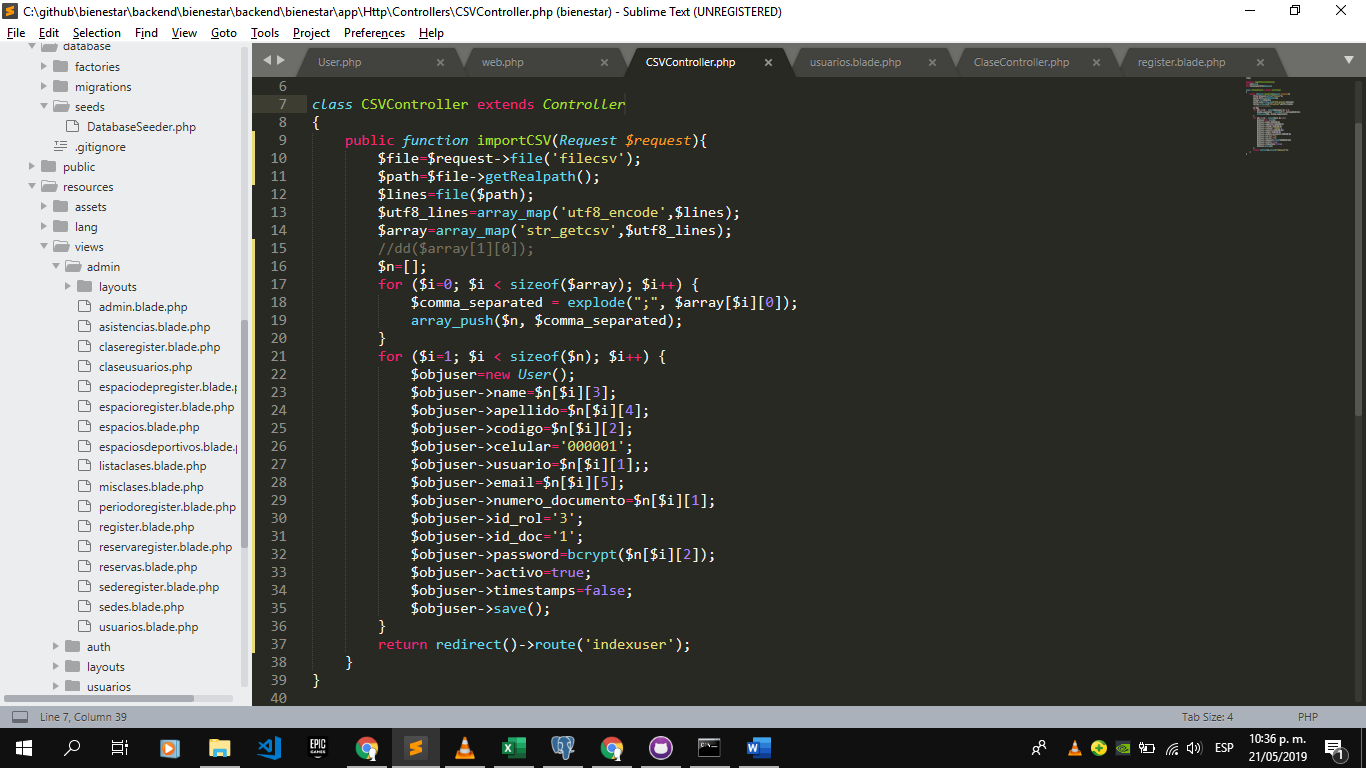


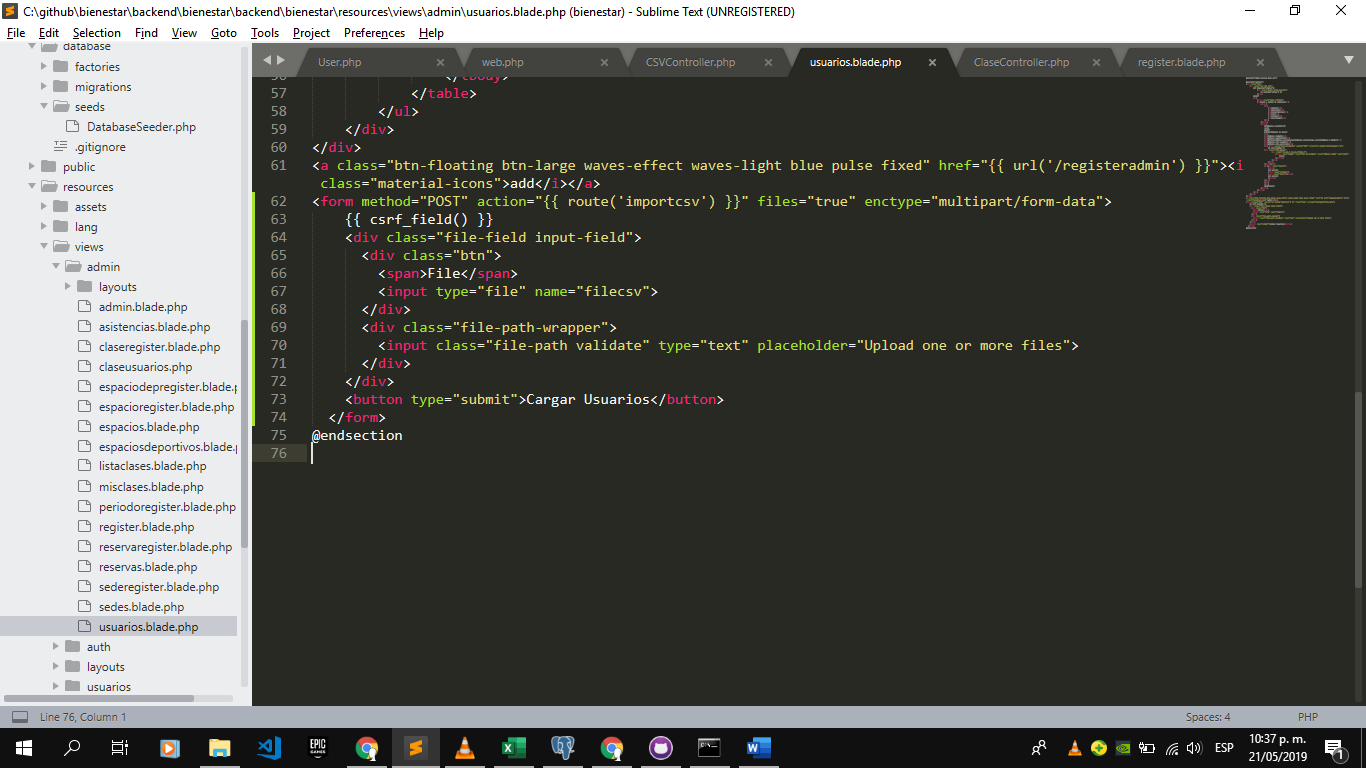




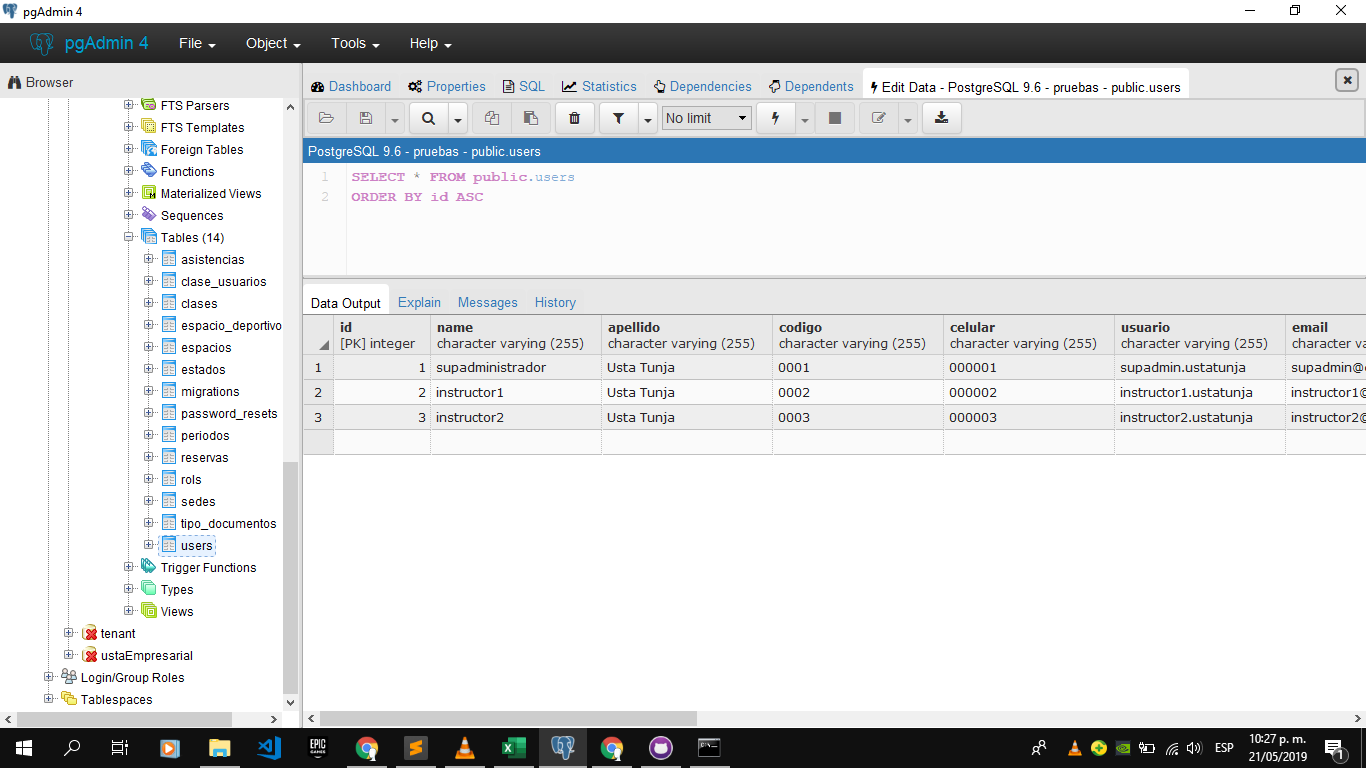
# Carga de usuarios al sistema por archivo plano (csv)

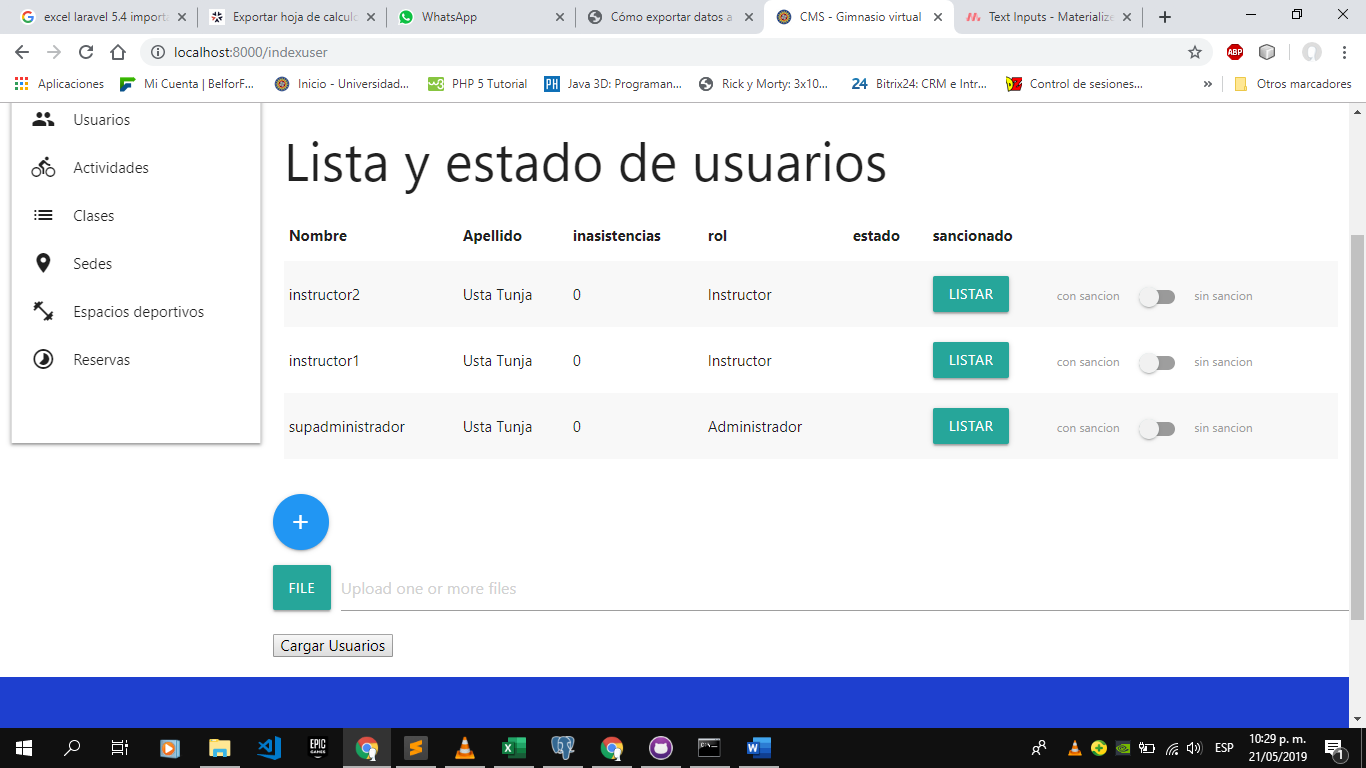
## Funcionamiento backend

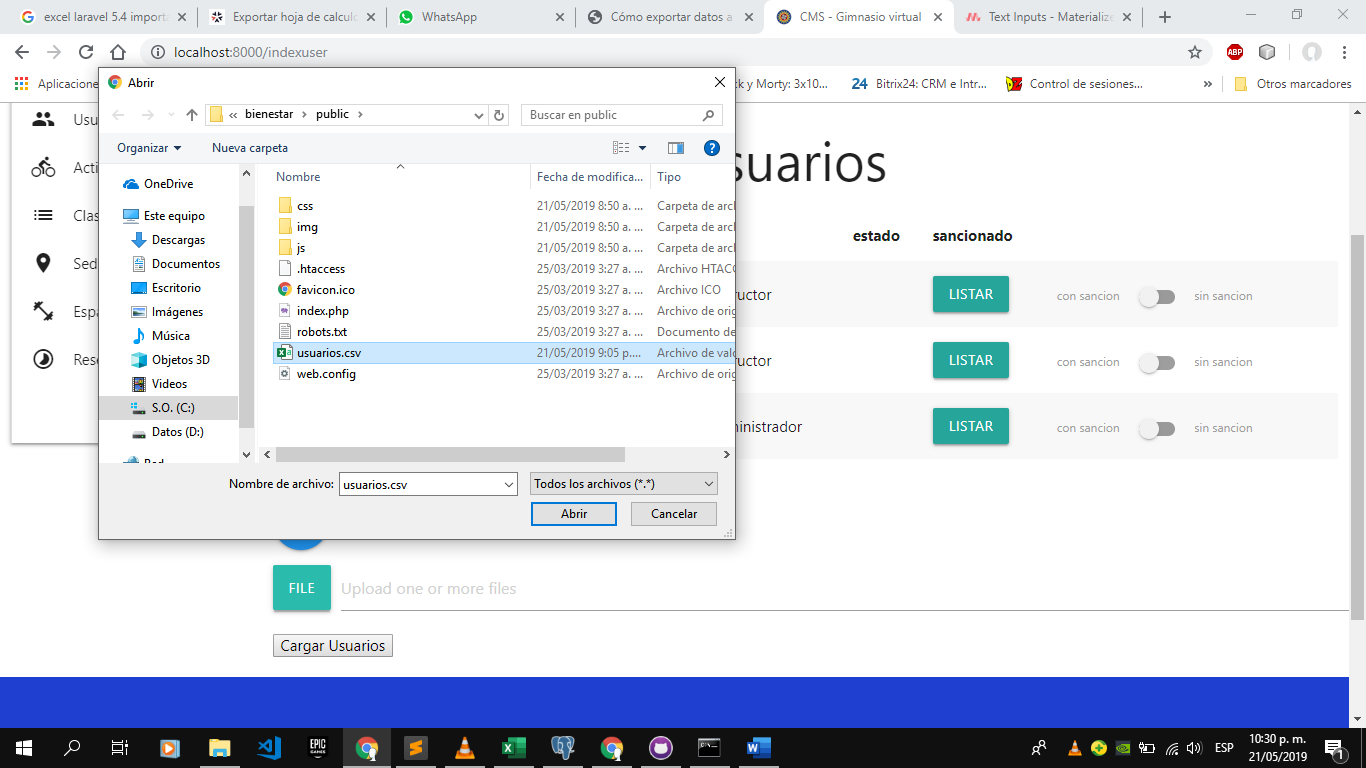


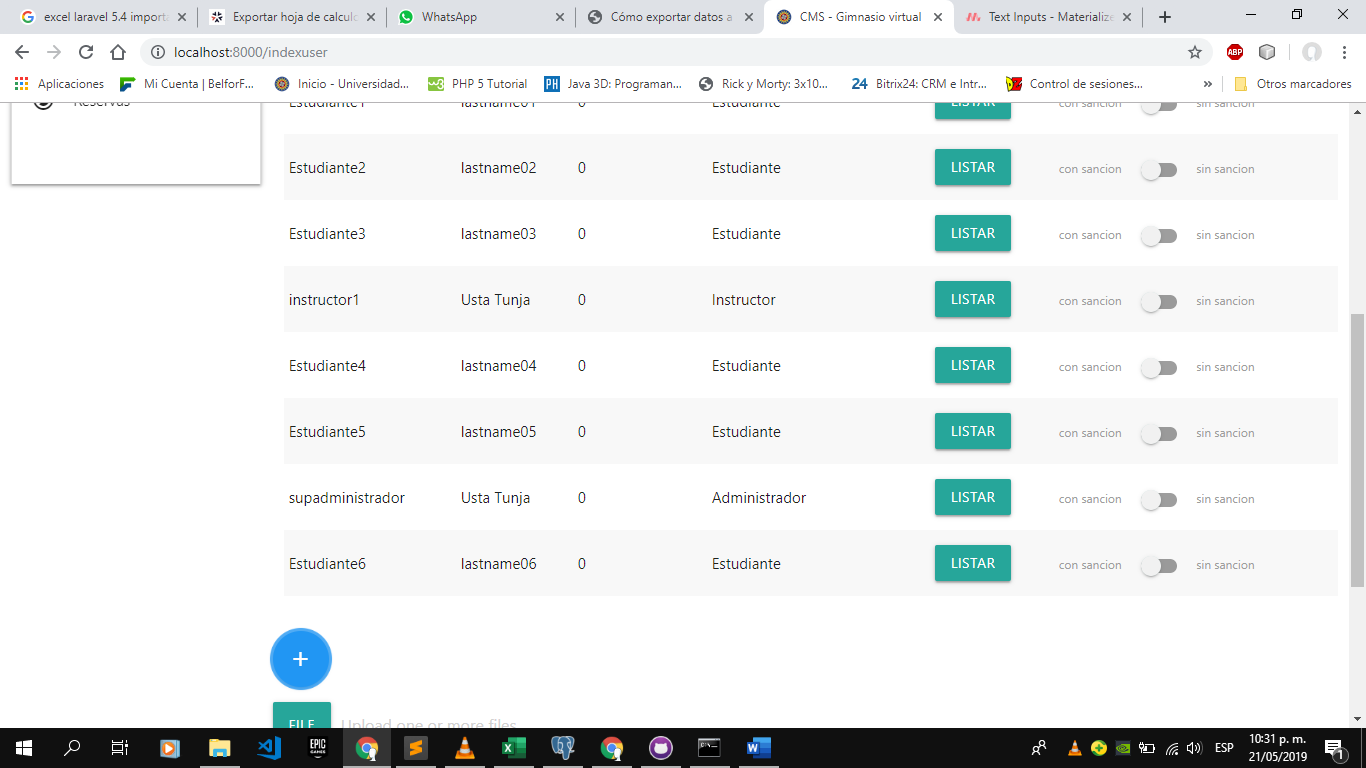


## Vista cms





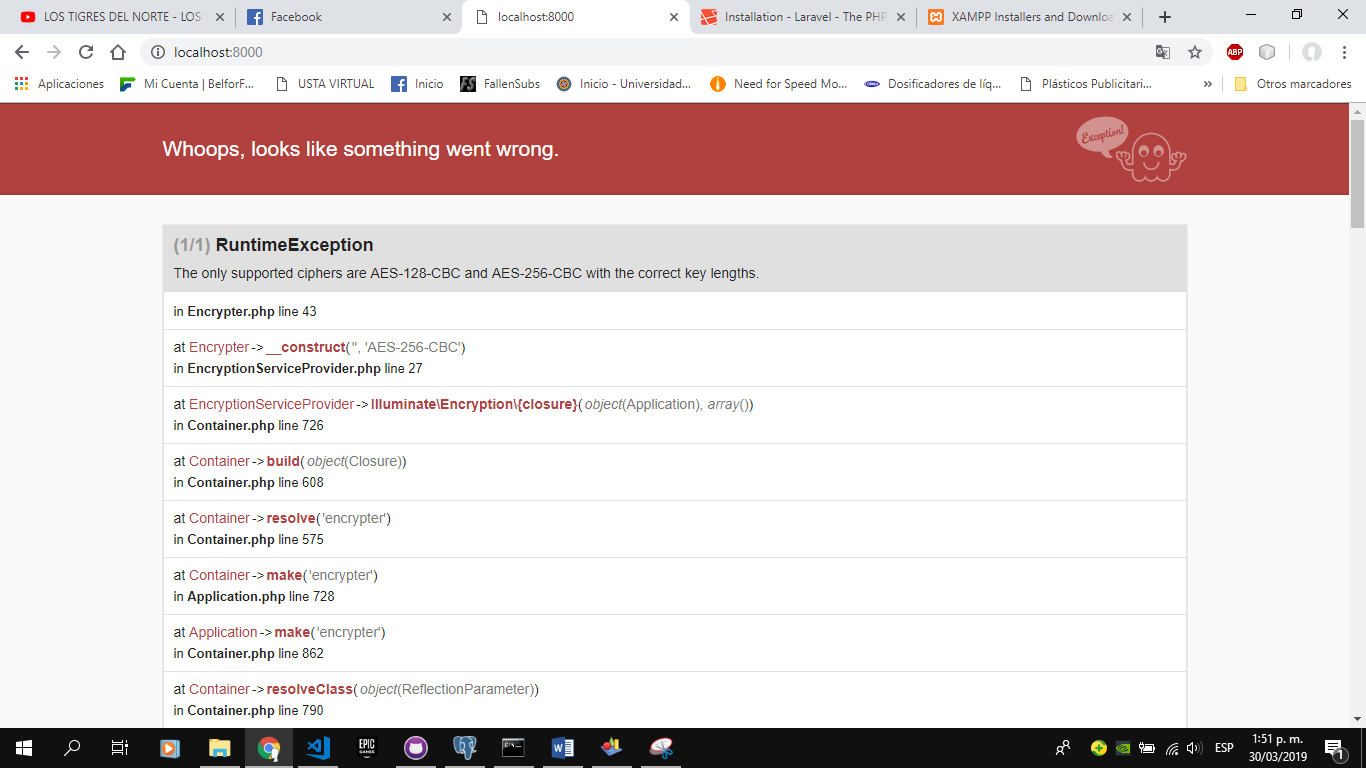


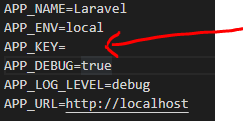


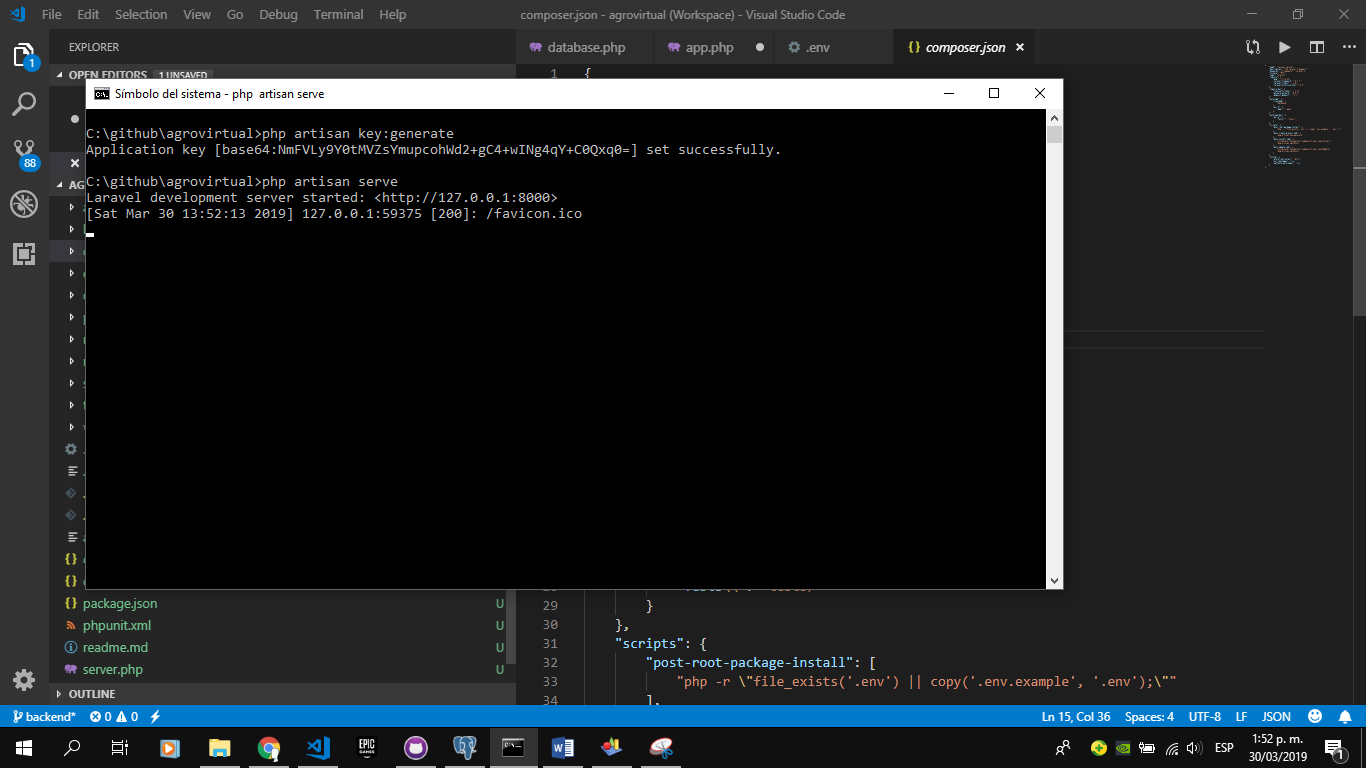
# Manejo de errores

## Runtime Exception:

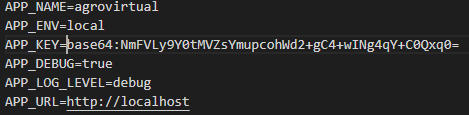
Si se presenta este error puede ser por dos razones. Primera no existe la llave “key”del proyecto, segunda, no existe el archivo .**env** en el proyecto.



Si existe el archivo .env, el key debe estar vacio y genera este error. 

En caso tal se soluciona mediante el comando php artisan key:generate este creara una llave para el proyecto 

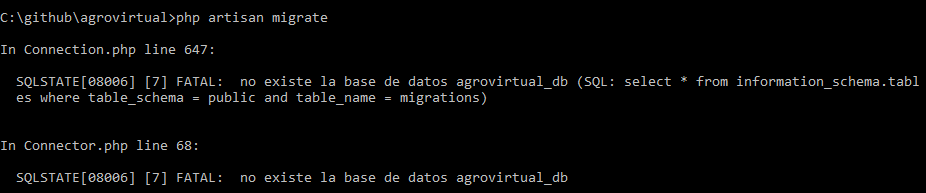
Cuando se genere la llave nustro archivo de configuracionquedara algo asi



Si el motivo es el segundo error, simplemente copiamos el archivo **.env.examle** y al nuevo archivo lo dejamos como .env en este archivo es donde esta la configuración del proyecto en cada equipo, por lo que cada copia del proyecto debe llevarlo y configurarlo según el equipo donde se instale el proyecto.

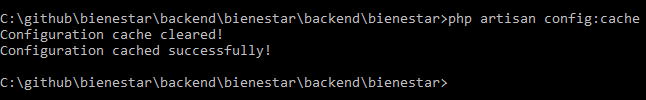
Luego de copiar este archivo se debe generar la llave ”key” como se sugirió en el primer error.

## Error en la migración

Un error que puede ocurrir durante la migración es el siguiente

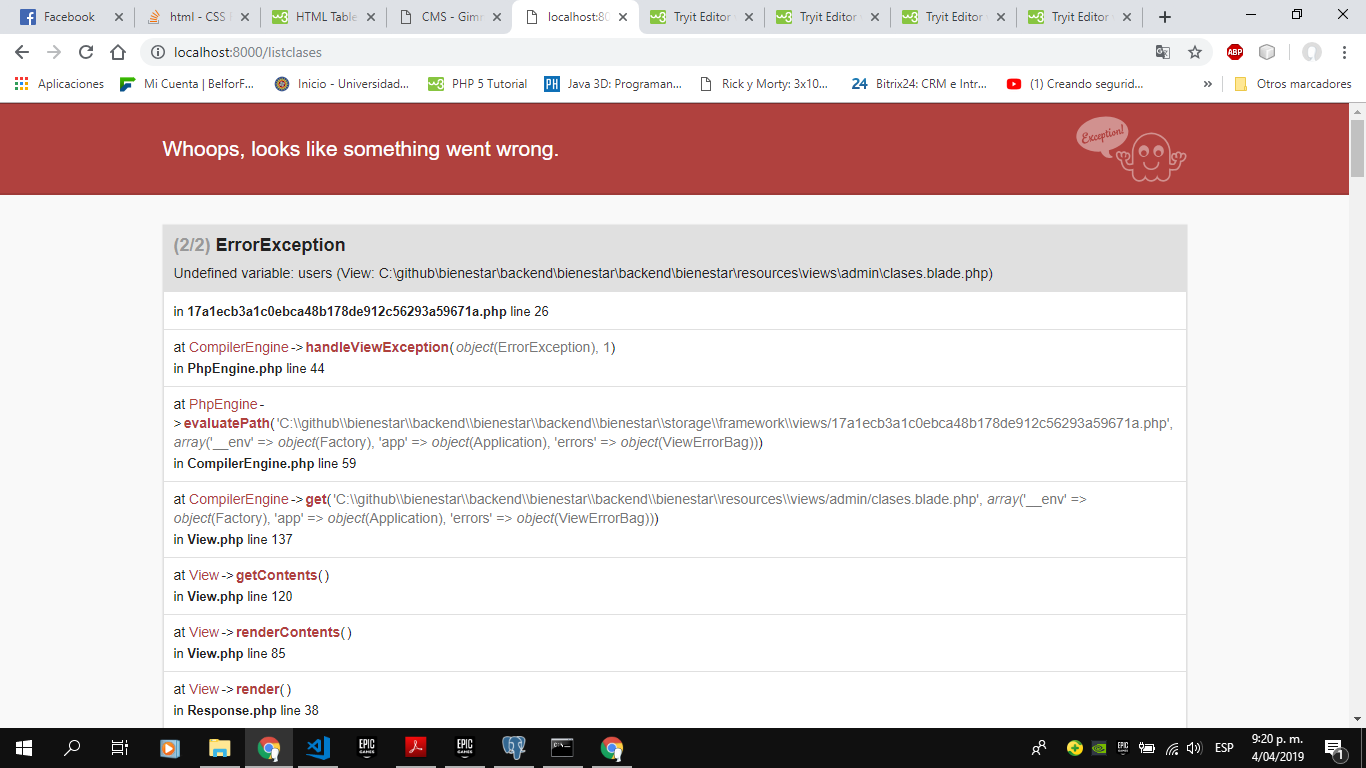
Esto sucede por que hay datos guardados en la configuración y en la cache que no se han actualizado. Esto se soluciona eliminado la cache mediante la siguiente línea de comandos

**Php artisan config:cache** esto limpia

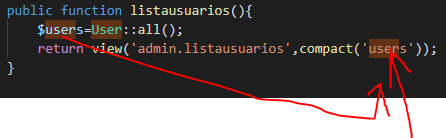


## Error user no encontrado

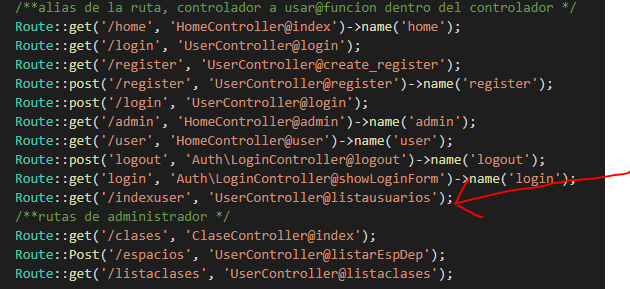
Este error se presenta por que no se reconocen los datos que pide la vista, es un error en el controlador

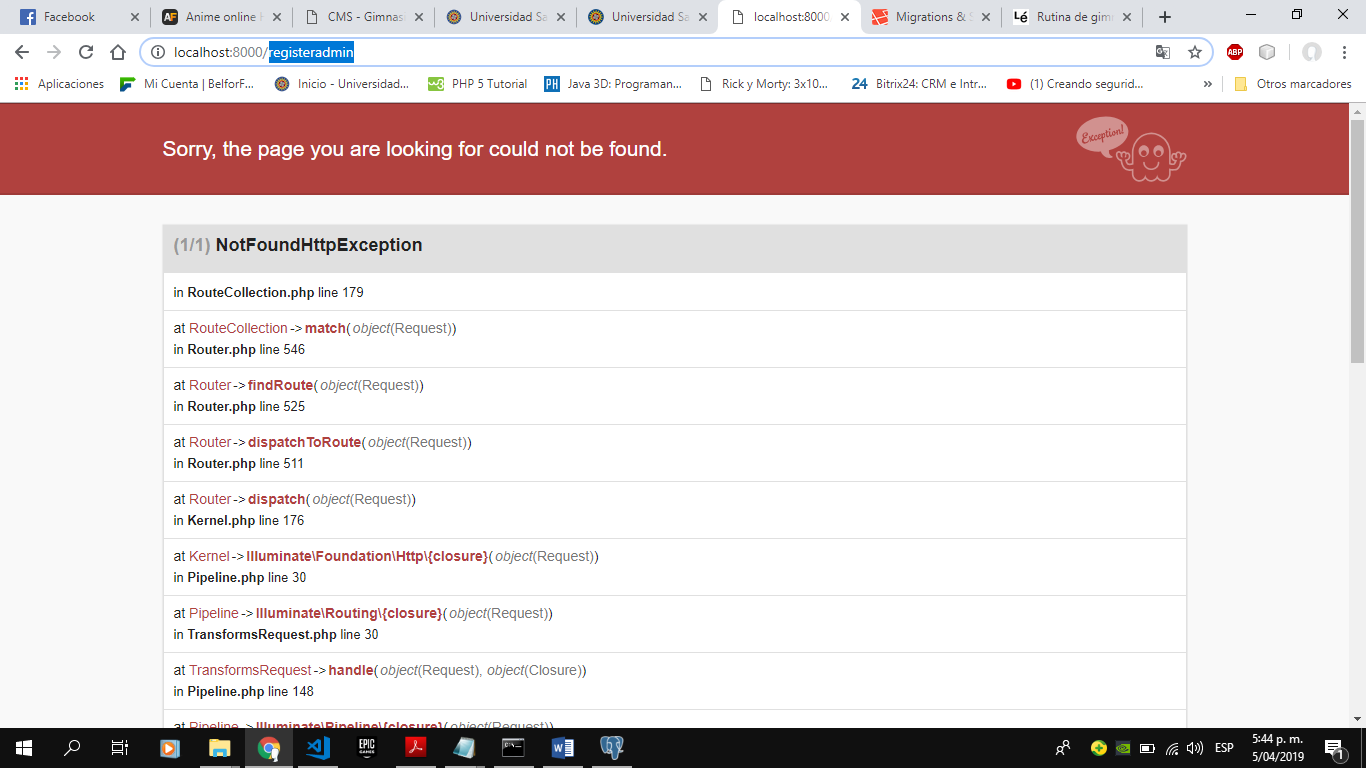


Este error se presenta por que no estamos enviando los datos desde el controlador a la tabla, se corrige enviando los datos por parámetro a la vista



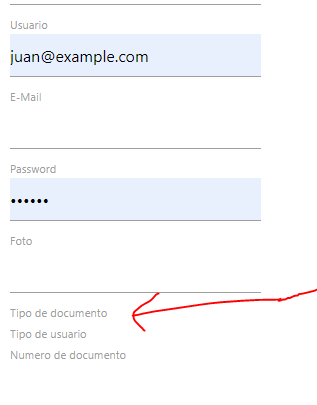
Llamado del controlador desde web.php



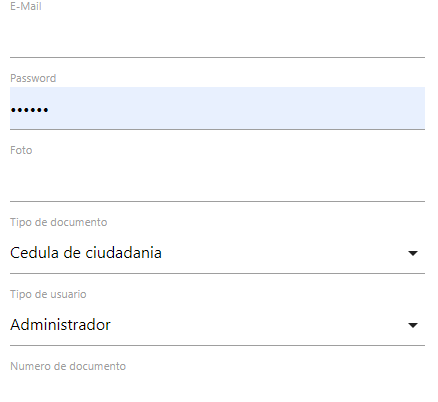
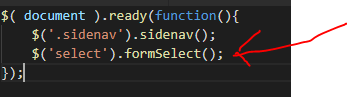


Verificar que la ruta exista, que tenga su método GET que trae el formulario y su método POST que envía los datos para ser guardados

## No despliega las listas de opciones

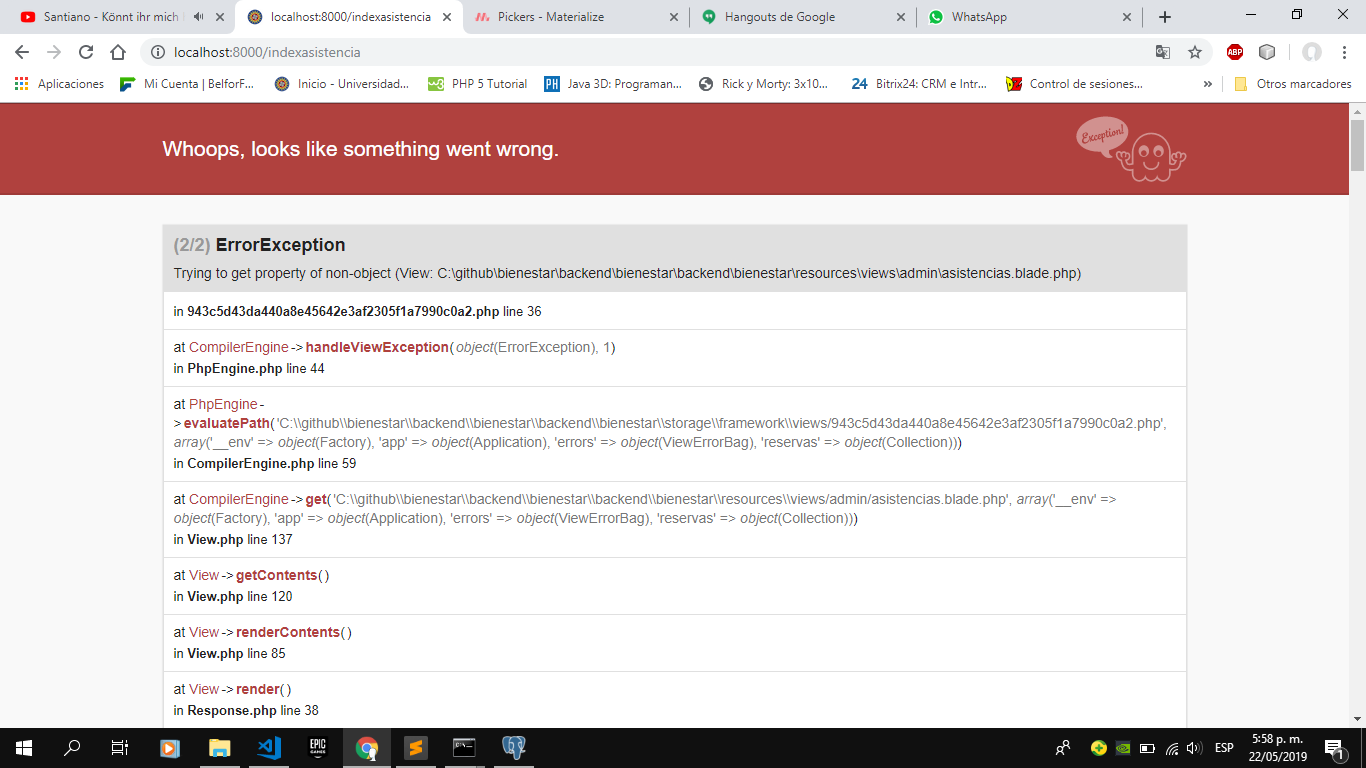


Falta implementación de jquery de matrialize para opciones de selección

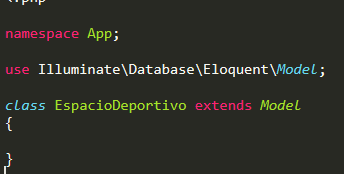


## Error de lectura de base de datos

Problema de obtención de consultas de base de datos, generado por falta de definición en el modelo o error en la base de datos.



Error en el modelo de Espacio deportivo

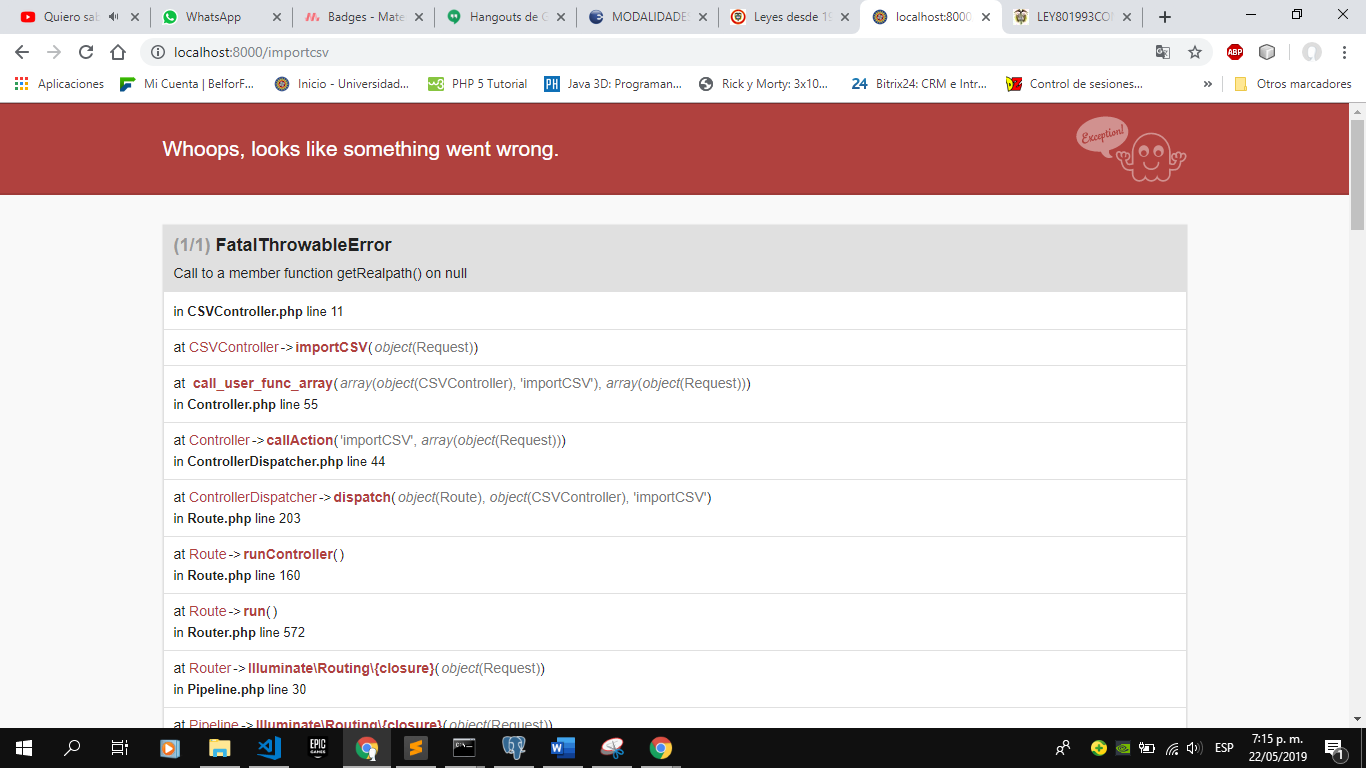


Solución:



## Error carga de usuarios

Este error se debe a que dio la orden de cargar archivo sin seleccionar un archivo para craga o el archivo no es valido.



Se soluciono validando el error por backend para que simplemente redirija al index de usuario cuando ocurra el error.

