

UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA

ING. SANTIAGO MENDEZ

Desarrollo Web

Sección D

## **PROYECTO FINAL – TORNEO NAVIDEÑO**



Bryan Josué Román García / 5190-21-1202

Abner Ottoniel López Tobar / 5190-17-1390

## MANUAL TECNICO DE PUBLICACION

### 1. PRE REQUISITOS:

- a. Para poder instalar el software de Laravel debemos de contar con una versión de PHP 8.3 o superior
- b. Un servidor Web (Apache o Nginx)
- c. También debemos de contar con composer (Administrador de dependencias de PHP)
- d. Una base de Datos (MySQL)
- e. NPM a través de Node.js

### 2. INSTALACION DE PHP, COMPOSER, NODE.JS Y LARAVEL

#### i. PHP

1. Instalamos xampp o laragon a través de su sitio web oficial.
2. Debemos de instalar la versión de PHP mas reciente desde su sitio web oficial, en este caso utilizaremos la v8.3.12.
  - a. Agregamos el archivo en nuestro directorio.
  - b. Agregamos un nuevo path con la ruta del PHP

#### ii. COMPOSER

1. Descargamos el archivo en su sitio web oficial y lo instalamos.

#### iii. NODE.JS

1. Descargamos el archivo a través de su sitio web oficial y lo instalamos.

#### iv. LARAVEL

1. Ingresamos al directorio en donde queremos instalar o crear nuestro proyecto de Laravel y abrimos una terminal.
2. Agregamos el siguiente comando: `laravel new nombre_proyecto --jet`
3. Configuramos nuestro entorno de instalación.
4. Luego de configurar el setup de instalacion del proyecto debemos de ejecutar los siguientes comandos en nuestra terminal:
  - a. `npm install && npm run dev`
    - i. `npm install` sirve para instalar todas las dependencias de Node.
    - ii. `Npm run dev` compila y construye todos los archivos css y js.
  - b. `php artisan serve`: Ejecuta el servidor

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\proyectos> laravel new proyecto-final --jet

Laravel

Which Jetstream stack would you like to install? [Livewire]:
[livewire] Livewire
[inertia ] Vue with Inertia
> 0

[ERROR] Value "0" is invalid

Which Jetstream stack would you like to install? [Livewire]:
[livewire] Livewire
[inertia ] Vue with Inertia
> livewire
```

```
Windows PowerShell

Would you like any optional features? [None]:
[none ] None
[api ] API support
[dark ] Dark mode
[verification] Email verification
[teams ] Team support
> none

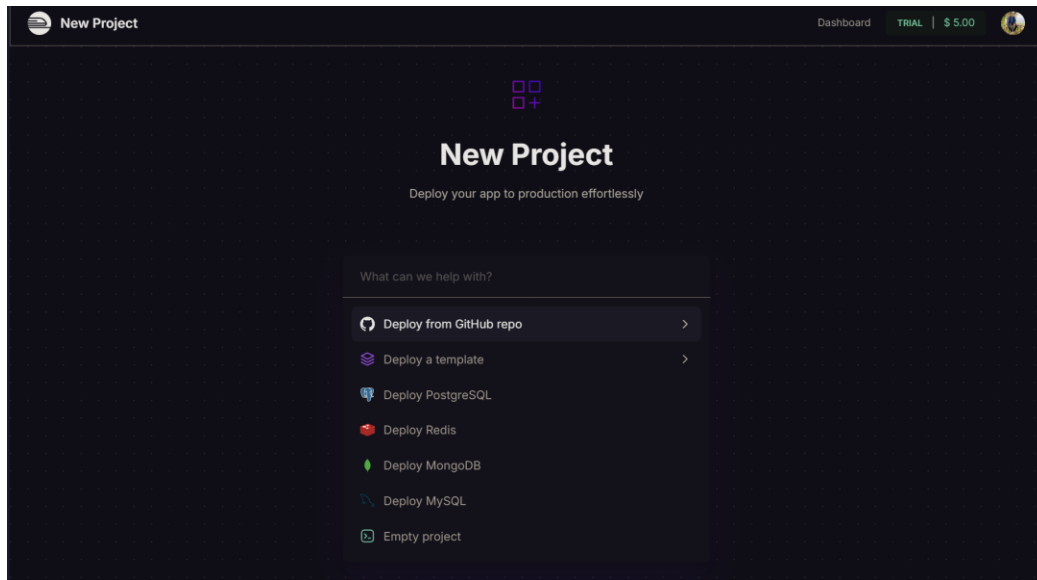
Which testing framework do you prefer? [Pest]:
[0] Pest
[1] PHPUnit
> 1
```

```
Which database will your application use? [SQLite]:
[sqlite ] SQLite
[mysql ] MySQL
[mariadb] MariaDB
[pgsql ] PostgreSQL (Missing PDO extension)
[sqlsrv ] SQL Server (Missing PDO extension)
> mysql

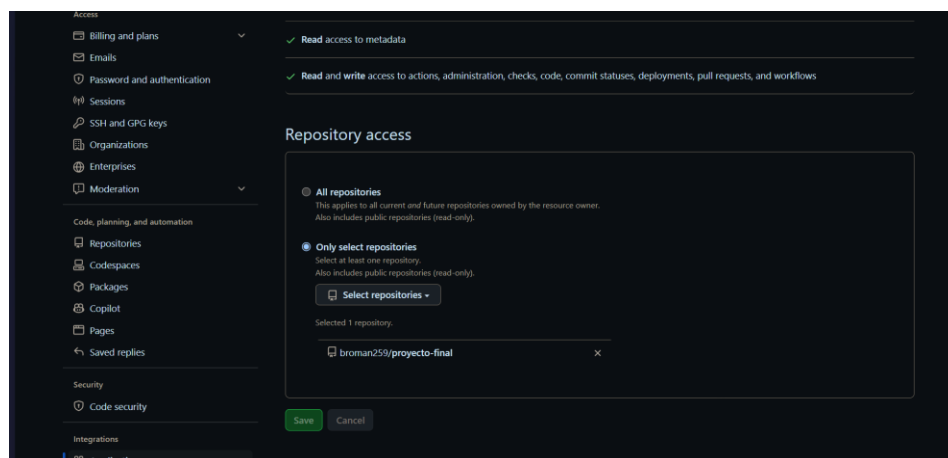
Default database updated. Would you like to run the default database migrations? (yes/no) [yes]:
> no
```

### 3. PUBLICACION WEB

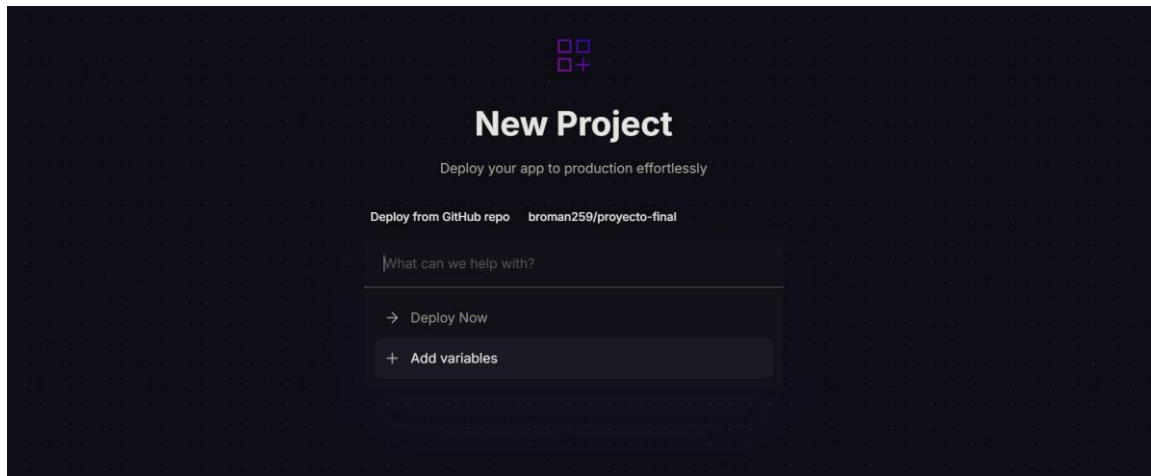
- a. Como primer paso debemos de subir nuestro proyecto a un repositorio de GitHub
- b. Ingresamos a Railway: <https://railway.app/>
- c. Iniciamos sesión con nuestra cuenta de GitHub y aprobamos los términos y condiciones
- d. Nos dirigimos a Create New Project y seleccionamos “Deploy from GitHub repo”



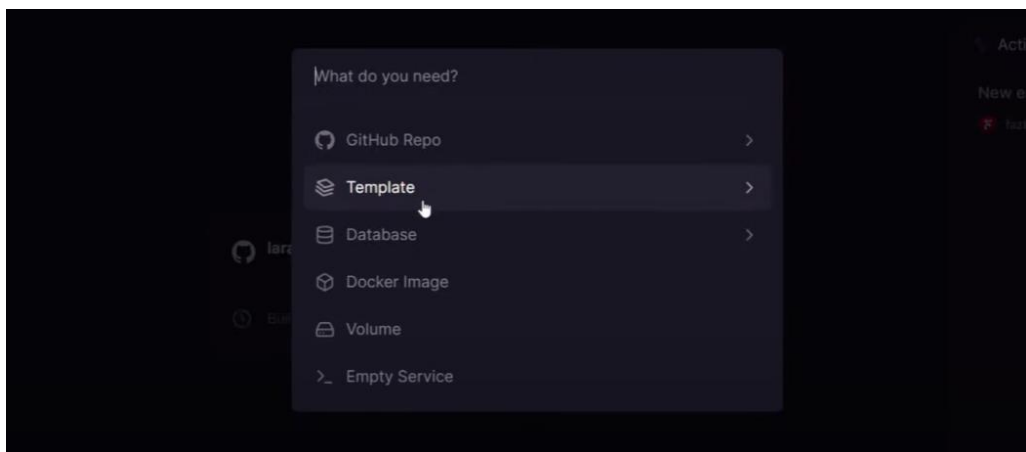
- e. Luego Indicamos que queremos configurar una nueva GitHub app
- f. Seleccionamos la cuenta de GitHub en donde se encuentre nuestro proyecto y aceptamos los permisos.
- g. Luego seleccionamos el repositorio en donde se encuentra nuestro proyecto y damos en continuar.



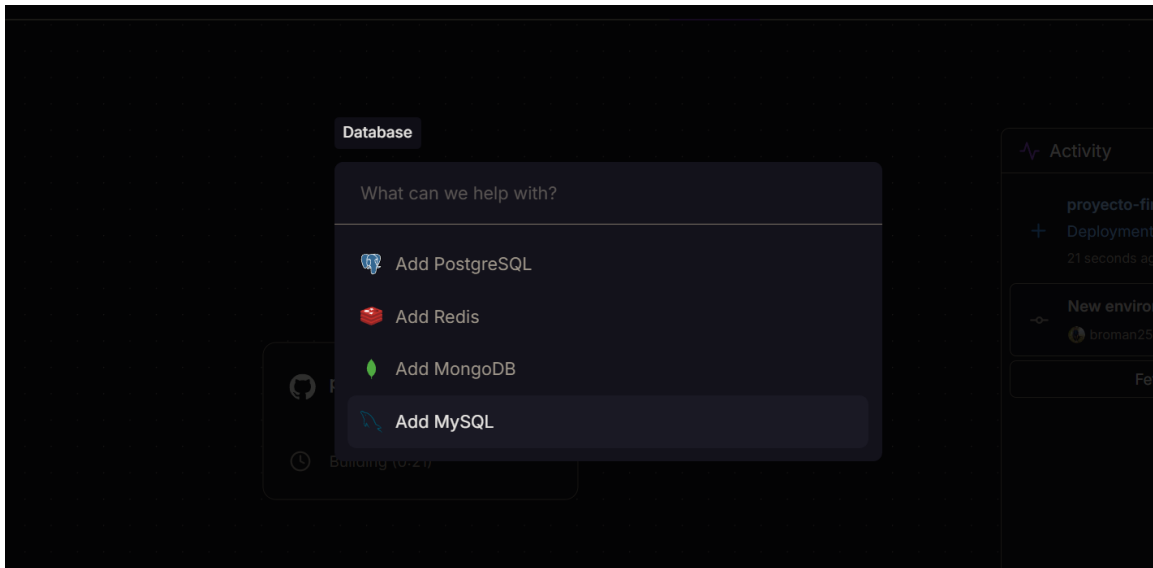
- h. Luego nos vamos al paso D y nos aparecerá nuestro proyecto. Lo seleccionamos y presionamos en donde dice Deploy Now



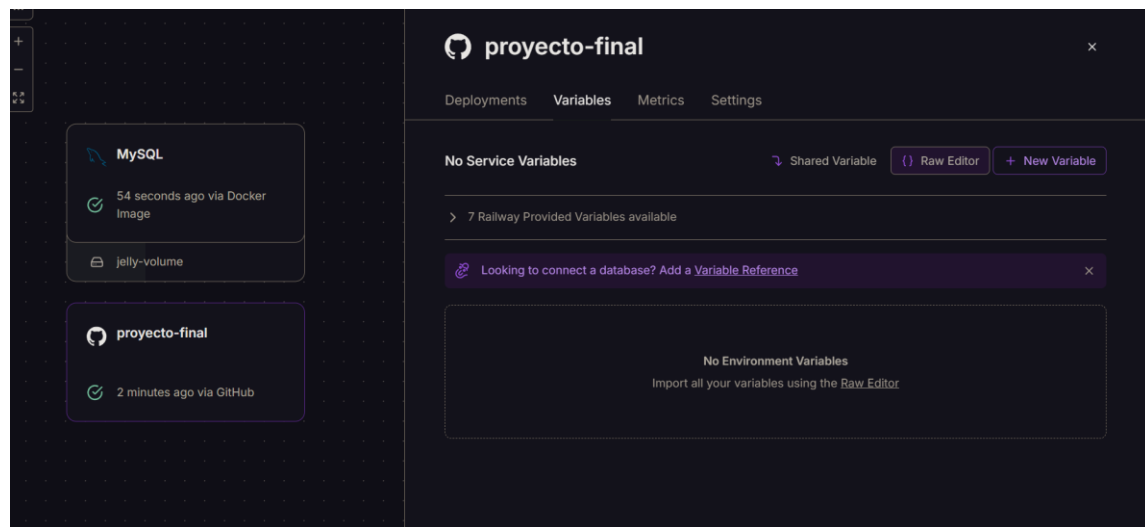
- i. Mientras se despliega nuestro proyecto tenemos que crear nuestra base de datos.
- j. Para ello nos vamos a donde dice New y seleccionamos Database



k. Elegimos la base de datos MySQL

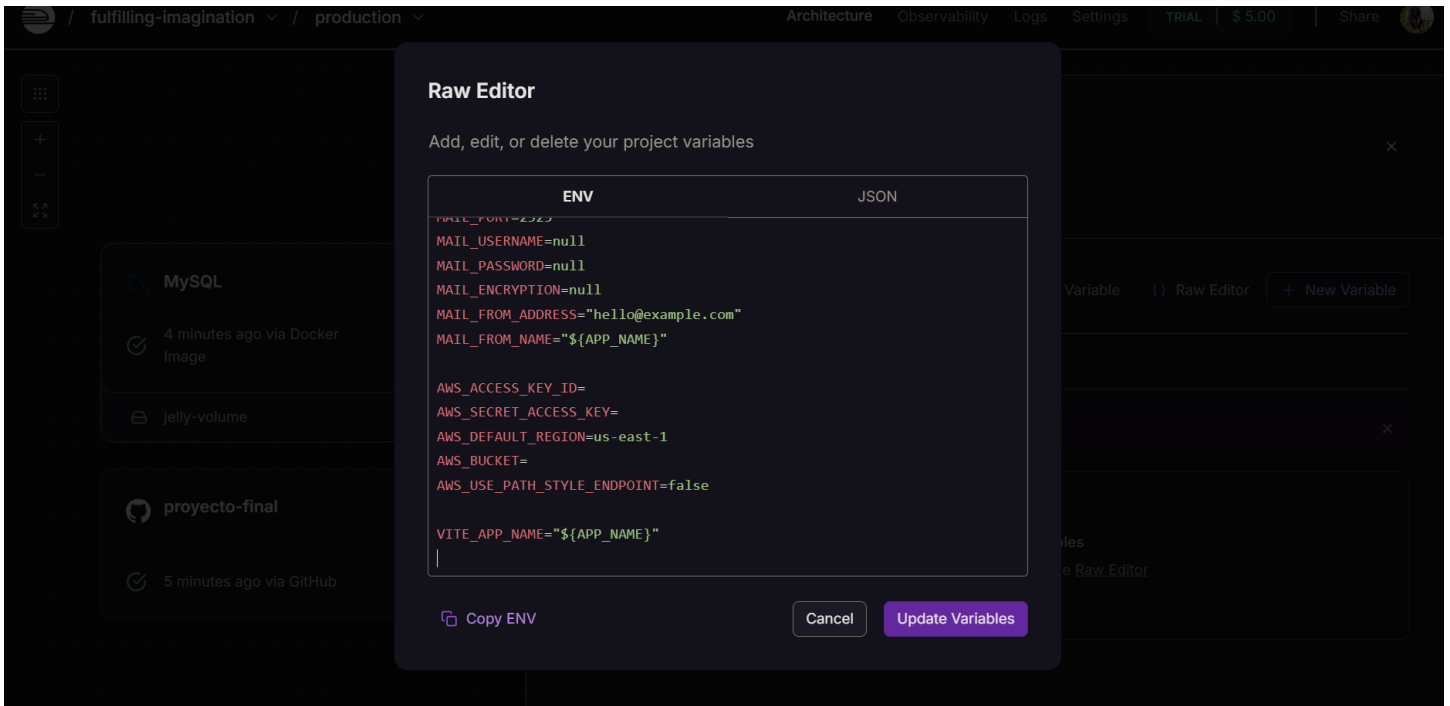


l. Luego entramos a configuracion del proyecto, nos vamos a variables y adentro de variables encontramos una opción llamada Raw Editor. Seleccionamos esa opción.

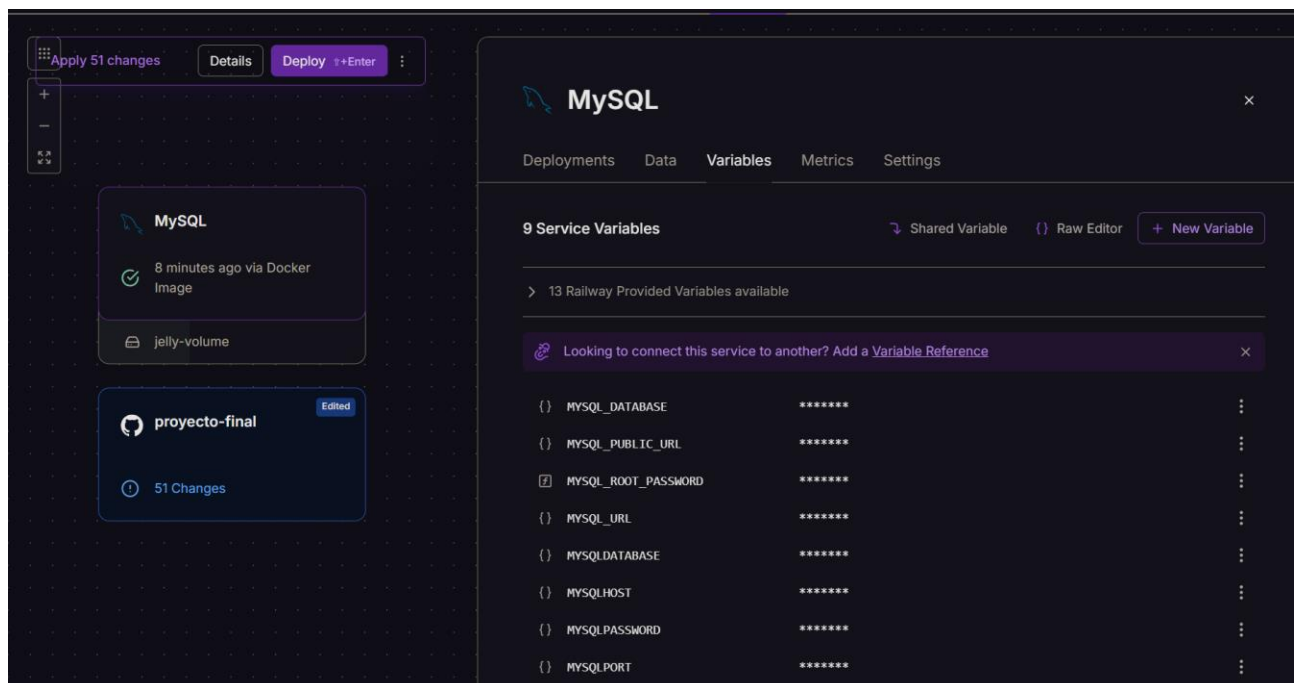


m. Nos dirigimos a nuestro archivo en nuestro VS CODE o editor de preferencia y abrimos el archivo .env (se encuentra casi al final) y copiamos todo el código que aparece en ese archivo.

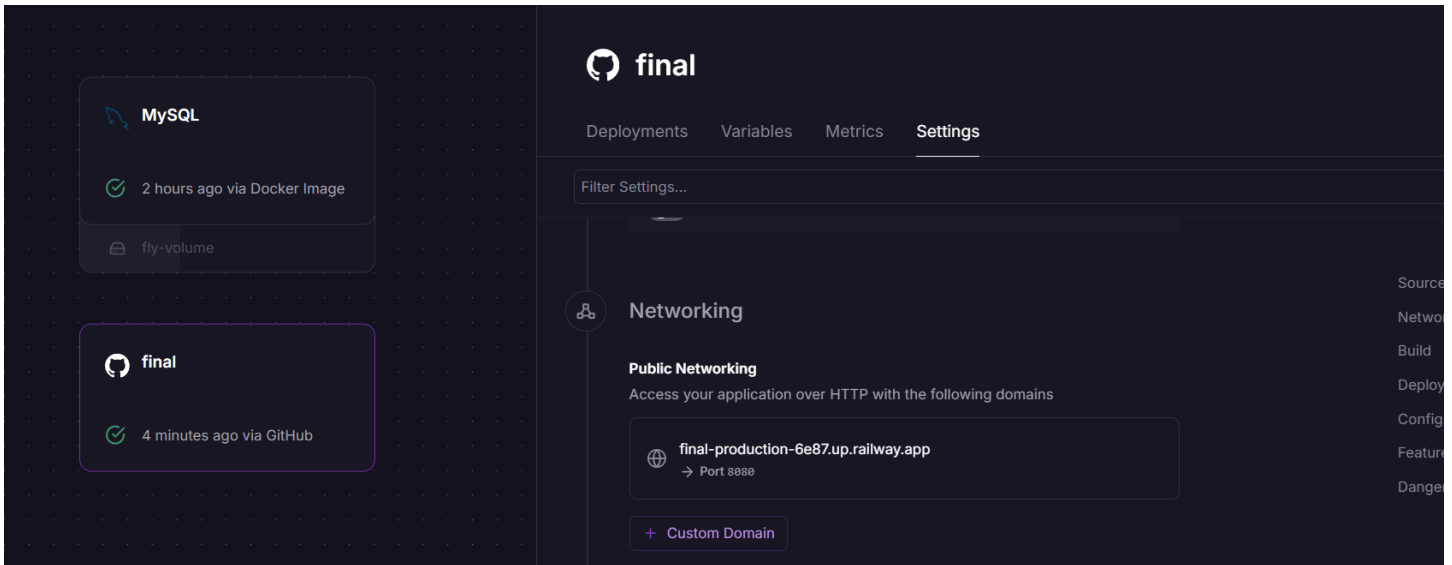
- n. Luego lo pegamos en nuestro Raw Editor en la opción de ENV



- o. Antes de guardar los cambios, nos vamos a la parte de DATABASE y debemos de cambiar nuestras propiedades de host por los datos que nos proporciona Railway. También debemos de cambiar la url de nuestro proyecto que por defecto esta en localhost.
- p. Para ello, guardamos cambios y seleccionamos nuestra Base de Datos en Railway y nos dirigimos al apartado de variables. Aquí es en donde copiamos los datos de nuestra DB hacia nuestro archivo ENV de nuestro proyecto.



- q. Cambiamos los datos de:
- i. DB HOST
  - ii. DB PORT
  - iii. DB PASSWORD
  - iv. DB USER
  - v. DB DATABASE (= MYSQL DATABASE EN RAILWAY)
- r. Luego nos dirigimos a nuestro proyecto y en el apartado de settings, en el apartado de Networking activamos la opción que dice Generate Domain.

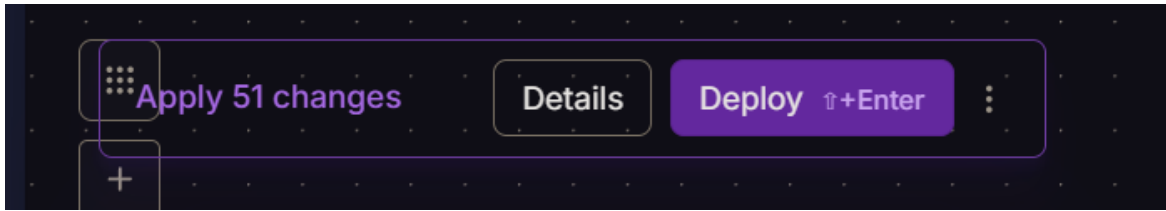


- s. Ahora copiamos esa url única que nos dio y la pegamos en nuestro archivos de variables ENV en el apartado de APP\_URL. (no se te olvide colocar https:// antes de la url que te dio railway).

```
APP_KEY="base64:0xwQAx9RnYXn8oug8envf01TL9VStRiJnHTsDKWsvlw="
APP_DEBUG="true"
APP_TIMEZONE="UTC"
APP_URL="https://final-production-6e87.up.railway.app"
APP_LOCALE="en"
```



t. Y ahora seleccionamos la opción que dice Deploy

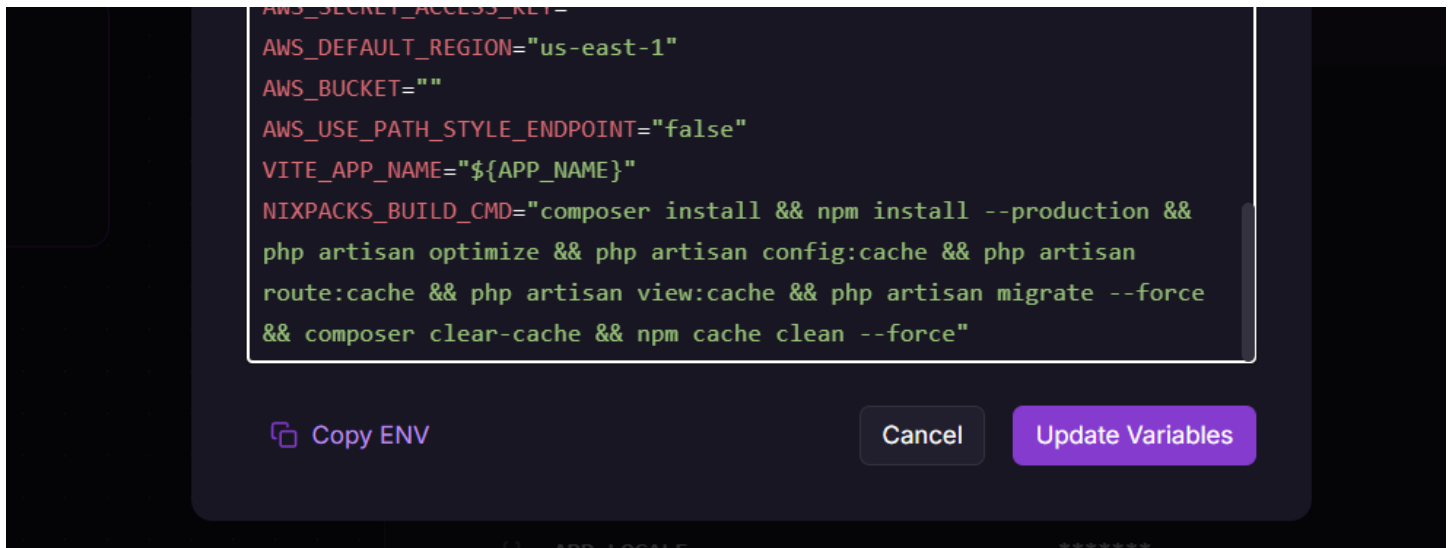


u. Tenemos la configuracion lista.

v. Ahora tenemos que escribir un comando que se ejecute cada vez que se despliega laravel para que actualice la base de datos, se pueda conectar correctamente o actualizar el código.

w. Para ello me voy al proyecto y me voy al apartado de variables y RAW EDITOR y al final, agrego lo siguiente:

i. NIXPACKS\_BUILD\_CMD="composer install && npm install --production && php artisan optimize && php artisan config:cache && php artisan route:cache && php artisan view:cache && php artisan migrate --force && composer clear-cache && npm cache clean --force"



x. Luego, hacemos un deploy como se muestra en el punto T y esperamos a que este activo nuestro deploy.

#### 4. PASOS UTILIZADOS PARA HACER USO DEL FRAMEWORK LARAVEL

- a. Para ello debemos de tener instalado todo lo anteriormente mencionado. Si deseas agregar el proyecto final a tu directorio puedes clonarlo a través del repositorio de GitHub.
  - i. <https://github.com/broman259/final>
- b. Una vez clonado el proyecto debes de abrir una terminal en la ruta del proyecto y ejecutar
  - i. `npm install` (para instalar las dependencias de Node en nuestro archivo `package.json`)
  - ii. `npm run dev`
  - iii. `php artisan serve`
- c. Enciende tu servidor APACHE y MySQL a través de (laragon o xampp)
- d. Luego debes de realizar las migraciones a la base de datos
  - i. En tu terminal ingresa el siguiente comando: `php artisan migrate`
  - ii. Una vez realizadas las migraciones procederemos a revisar si nuestras tablas se crearon correctamente en nuestro servidor de Base de Datos. (phpmyadmin)
  - iii. Una vez realizado todo lo anterior procederemos a revisar nuestro proyecto en nuestro sitio web local a través de la dirección <http://127.0.0.1:8000> o en la dirección que te aparezca al ejecutar `php artisan serve`.

#### 5. RECOMENDACIONES PARA EL USO DEL FRAMEWORK LARAVEL

- a. Para poder utilizar laravel es necesario tener conocimientos en HTML, CSS, JS, GIT (no es necesario)
- b. Se recomienda tener la versión mas actualizada para un mejor rendimiento.
- c. Utilizar las técnicas y estructuras recomendadas, así como usar controladores, modelos, rutas, migraciones, seeders, entre otras.

#### 6. COMENTARIOS PERSONALES

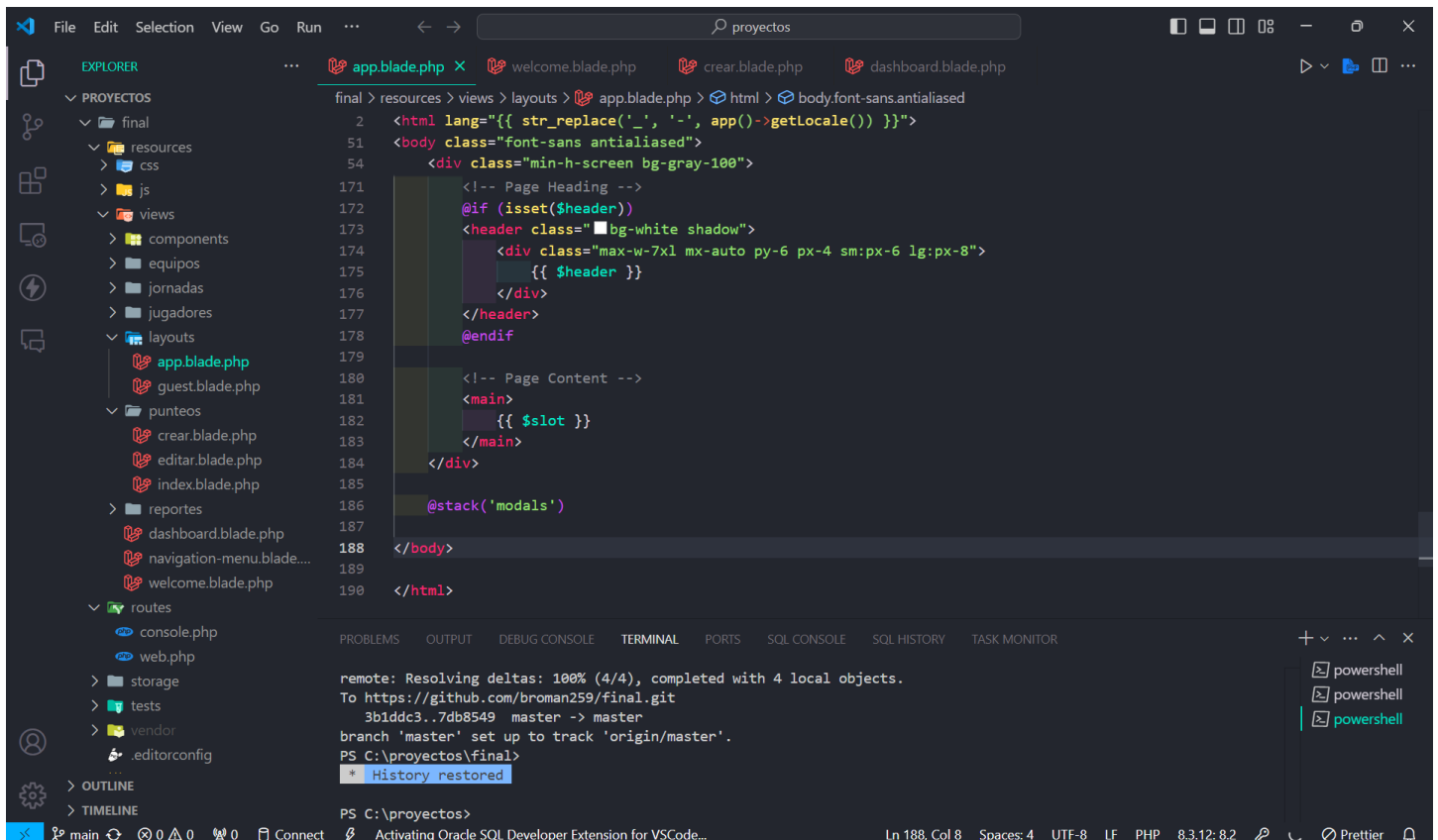
- a. Considero que si no estas familiarizado con ningún framework similar puede resultarte algo tedioso, ya que al inicio puedes ver demasiadas carpetas estructuradas, pero al final reduce bastante el trabajo a comparación con realizarlo con PHP puro.
- b. El uso de migraciones, modelos y controladores en lo personal me ayudo y me facilito demasiado el trabajo para la creación del proyecto.

- c. Laravel es muy reconocido y muy usado, así que hay demasiada documentación acerca de este framework y eso facilita mucho la solución a errores.
- d. Al principio si fue algo tedioso el ver tantas carpetas y no se sabe por donde iniciar, pero una vez comienzas a ver la estructura te das cuenta de que es mas entendible y fácil de localizar todo a comparación si lo haces en un solo archivo.

## 7. CODIGO FUENTE

- a. A continuación, dejare algunas imágenes como prueba del proyecto realizado.
- b. También pueden ver el proyecto en mi repositorio de GitHub:

<https://github.com/broman259/final>



The screenshot shows a Visual Studio Code editor interface. On the left, the Explorer sidebar displays a project structure for 'PROYECTOS' with a 'final' directory containing 'resources', 'views', 'components', 'equipos', 'jornadas', 'jugadores', 'layouts', 'punteos', 'reportes', 'storage', 'tests', and 'vendor'. The 'layouts' directory is expanded, showing 'app.blade.php', 'guest.blade.php', 'crear.blade.php', 'editar.blade.php', 'index.blade.php', 'dashboard.blade.php', 'navigation-menu.blade.php', and 'welcome.blade.php'. The main editor area displays the content of 'app.blade.php' in the 'resources > views > layouts' path. The code is a Blade template with HTML structure, including a header section with a class 'bg-white shadow' and a main content area. The terminal at the bottom shows a successful git pull operation from the repository 'https://github.com/broman259/final.git', resolving deltas and restoring history.

```
final > resources > views > layouts > app.blade.php > html > body.font-sans.antiased
2  <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
51  <body class="font-sans antiased">
54  <div class="min-h-screen bg-gray-100">
171  <!-- Page Heading -->
172  @if (isset($header))
173  <header class="bg-white shadow">
174  <div class="max-w-7xl mx-auto py-6 px-4 sm:px-6 lg:px-8">
175  {{ $header }}
176  </div>
177  </header>
178  @endif
179
180  <!-- Page Content -->
181  <main>
182  {{ $slot }}
183  </main>
184  </div>
185
186  @stack('modals')
187
188 </body>
189
190 </html>
```

remote: Resolving deltas: 100% (4/4), completed with 4 local objects.  
To https://github.com/broman259/final.git  
3b1ddc3..7db8549 master -> master  
branch 'master' set up to track 'origin/master'.  
PS C:\proyectos\final>  
\* History restored



DashboardEquiposJugadoresJornadasPunteosRep. JugadoresRep. Jugadores\*EquipoRep EquiposRep. Anotadores

Dashboard

Bienvenido al Dashboard

Torneo Navideño de BasketBall LA CANASTA

BRYAN JOSUE ROMAN GARCIA

5190-21-1202

ABNER OTTONIEL LOPEZ TOBAR

5190-17-1390

Proyecto Final Desarrollo Web






Reporte General de Jugadores

Reporte General de Jugadores

Agrupados por Equipo

Exportar Jugadores

Buscar equipo o jugador...




Equipo	Jugador	Fecha de Nacimiento	Creado el	Última Actualización
 NY - Knicks11			22/10/2024 21:47:28	22/10/2024 21:47:28
	 jugador 2 A Jugador 2	2024-10-15	22/10/2024 21:48:14	23/10/2024 20:40:08
 Hornets			18/10/2024 09:53:07	18/10/2024 09:53:07
	 Jugador 1 A Jugador 1	2024-10-04	18/10/2024 09:53:56	23/10/2024 20:39:47
 NY - Knicks			18/10/2024 09:52:57	22/10/2024 19:07:47

Punteos

Lista de Punteos x Jornada

Agrega el punteo al jugador por cada jornada

Agregar Punteo

No.	Jornada	Jugador	Tipo de Tiro	Puntos Obtenidos	Creado el	Última Actualización	Acciones
1	Jornada 1	 Jugador 1 A Jugador 1	2 Puntos	2	18/10/2024 16:07:22	18/10/2024 16:07:22	<div>EDITAR</div> <div>ELIMINAR</div>
2	jornada 2	 jugador 2 A Jugador 2	3 Puntos	3	22/10/2024 21:49:16	22/10/2024 21:49:16	<div>EDITAR</div> <div>ELIMINAR</div>
3	Jornada 1	 jugador 2 A Jugador 2	3 Puntos	3	23/10/2024 19:14:07	23/10/2024 19:14:07	<div>EDITAR</div> <div>ELIMINAR</div>


AutoSave Off equipos... Saved to this PC Search

File Home Insert Draw Page Layout Formulas Data Review View Automate Help

Clipboard Paste Font Alignment Number Styles Cells Editing Add-ins Analyze Data

Comments Share

A1 No.

No.	Nombre	Imagen	Fecha de Creación	Última Actualización
1	NY - Knicks		25-10-2024 04:31:42	25-10-2024 04:31:42
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				

Worksheet

## 8. ALGUNOS COMANDOS QUE PUEDEN SERVIR

- a. ejecutar nuestras migraciones
  - i. `php artisan migrate`
- b. crear una migracion
  - i. `php artisan make:migration create_jugadores_table --create=jugadores`
- c. crear un modelo
  - i. `php artisan make:model Modelo` (en singular)
- d. crear un controlador
  - i. `php artisan make:controller NombreController --resource`
- e. crear un exportable
  - i. `php artisan make:export UsersExport --model=User`

### f. INSTALACIONES ADICIONALES

- i. Exportable a excel
  - 1. `composer require maatwebsite/excel`
- ii. Exportable a PDF (no realizado en este proyecto)
  - 1. `composer require barryvdh/laravel-dompdf`

### g. COMANDOS ADICIONALES

- i. Limpia el caché del composer
  - 1. `composer dump-autoload`
- ii. Limpia el cache de configuración
  - 1. `php artisan config:clear`
- iii. Ejecución del servidor web de Laravel.
  - 1. `php artisan serve` (OPCION 1)
  - 2. `php -S 127.0.0.1:8000 -t public` (OPCION 2)
- iv. GIT
  - 1. `git init`
  - 2. `git add .`
  - 3. `git status`
  - 4. `git commit -m "inicializacion del proyecto"`

5. `git remote add origin https://github.com/broman259/final.git`
6. `git push -u origin master`

9. ENLACE DEL SITIO WEB

- a. <https://final-production-6e87.up.railway.app/>