

---

*Proyecto Intermodular*

---

**DESPLIEGUE DE BITKEYS**

**ALEJANDRO RICO SÁNCHEZ  
ADRIÁN GUTIÉRREZ GALVAÑ**

**27/02/2026**



# Índice

1. Instancia AWS.....	3
1.1. Grupo de seguridad.....	3
1.2. Instancia.....	3
1.3. IP Elástica.....	4
2. Usuarios.....	5
2.1. Usuarios, directorios y permisos.....	5
2.2. Mensaje de bienvenida SSH.....	5
3. Configuración SSH.....	6
3.1. Claves.....	6
3.2. Seguridad.....	7
4. Certificados para HTTPS.....	8
4.1. LetsEncrypt.....	8
5. Apache.....	9
5.1. Virtual Hosts.....	9
6. Servidor FTP.....	11
6.1. vsftpd.conf.....	11
6.2. vsftpd.userlist.....	12
6.3. Comprobación vsftpd.....	12
7. Despliegue de la aplicación.....	13
7.1. APP.....	13
7.2. Test.....	21
7.3. Backup.....	23

# 1. Instancia AWS

## 1.1. Grupo de seguridad

Reglas de entrada del grupo de seguridad web-ftp-project-sg, que se utilizará para la instancia EC2 que contenga la aplicación.

Type	Info	Protocol	Info	Port range	Info	Source	Info
SSH	▼	TCP		22		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X
HTTP	▼	TCP		80		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X
HTTPS	▼	TCP		443		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X
Custom TCP	▼	TCP		21		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X
Custom TCP	▼	TCP		20		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X
Custom TCP	▼	TCP		30000 - 30050		Anywhere	▼
						<input type="text"/> 0.0.0.0/0	X

## 1.2. Instancia

Creamos una instancia EC2 llamada Servidor-Web-AWS en la que ponemos ese grupo de seguridad y un par de claves .pem generado.

Instance summary for i-00ecdde0af1b9e7c (Servidor-Web-AWS)		Info	Connect	Instance state	Actions
Updated less than a minute ago					
Instance ID	<input type="text"/> i-00ecdde0af1b9e7c	Public IPv4 address	<input type="text"/> 98.94.9.204   open address		Private IPv4 addresses
IPv6 address	-	Instance state	<input checked="" type="radio"/> Running	<input type="text"/> 172.31.16.31	Public DNS
Hostname type		Private IP DNS name (IPv4 only)	<input type="text"/> ip-172-31-16-31.ec2.internal	<input type="text"/> ec2-98-94-9-204.compute-1.amazonaws.com   open address	Elastic IP addresses
Answer private resource DNS name	IPv4 (A)	Instance type	t2.medium	-	AWS Compute Optimizer finding
Auto-assigned IP address	<input type="text"/> 98.94.9.204 [Public IP]	VPC ID	<input type="text"/> vpc-072c1f5fcf4123208	<input type="radio"/> Opt-in to AWS Compute Optimizer for recommendations.   Learn more	
IAM role	-	Subnet ID	<input type="text"/> subnet-0a0c25d3c63372316		Auto Scaling Group name
IMDSv2	Required	Instance ARN	<input type="text"/> arn:aws:ec2:us-east-1:292873490331:instance/i-00ecdde0af1b9e7c	-	Managed
Operator	-				false

### 1.3. IP Elástica

Generamos una IP elástica y la asociamos con la instancia EC2. La IP de nuestro EC2 es: 3.213.82.57.

**Elastic IP address: 3.213.82.57**

**Resource type**  
Choose the type of resource with which to associate the Elastic IP address.

Instance  
 Network interface

⚠ If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the address will still be allocated to your account. [Learn more](#)

If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.

**Instance**  
 X C

**Private IP address**  
The private IP address with which to associate the Elastic IP address.

**Reassociation**  
Specify whether the Elastic IP address can be reassigned to a different resource if it is already associated with a resource.

Allow this Elastic IP address to be reassigned

## 2. Usuarios

### 2.1. Usuarios, directorios y permisos

Creamos los usuarios app y backup y la estructura de directorios necesaria. Para ello pondremos los siguiente comandos:

#### app

```
sudo adduser --disabled-password --gecos "" app
sudo mkdir -p /home/app/ftp/www
sudo mkdir -p /home/app/logs
sudo chown -R app:app /home/app/ftp/www
sudo chown -R app:app /home/app/logs
sudo chmod 755 /home/app/ftp/www
```

#### backup

```
sudo adduser --disabled-password --gecos "" backup
sudo mkdir -p /home/backup/ftp/fitxers
sudo chown -R backup:backup /home/backup/ftp/fitxers
sudo chmod 755 /home/backup/ftp/fitxers
```

#### test

```
sudo mkdir -p /home/app/ftp/test
sudo chown -R app:app /home/app/ftp/test
```

### 2.2. Mensaje de bienvenida SSH

Configuramos el mensaje de bienvenido del usuario ubuntu al conectarse por SSH a la instancia. Para ello editamos e archivo que se encuentra en la siguiente ruta:

```
sudo nano /etc/update-motd.d/00-header
```

```
ubuntu@ip-172-31-16-31:~$ 
GNU nano 7.2          /etc/update-motd.d/00-header *
#!/bin/sh
#
#      00-header - Mensaje de bienvenida personalizado para AWS
#
printf "\n"
printf "=====\\n"
printf "Benvingut a la instància de Servidor Web en AWS\\n"
printf "Grup: Adrián Gutiérrez y Alejandro Rico\\n"
printf "=====\\n"
printf "\\n"
```

Nos desconectamos y volvemos a conectar para ver si funciona correctamente

```
batoi@AlejandroRico:~$ ssh -i '/home/batoi/Documentos/Segundo/PI/despliegue/PI-a
lejandro.pem' ubuntu@3.213.82.57
=====
Benvingut a la instància de Servidor Web en AWS
Grup: Adrián Gutiérrez y Alejandro Rico
=====
```

### 3. Configuración SSH

#### 3.1. Claves

Añadimos las tres claves públicas (Alejandro, Adrián y Rubén) al archivo de configuración authorized\_keys para que podamos conectarnos a la instancia.

Para generar las claves públicas se han extraído a partir de los archivos .pem que contienen la clave privada con el comando ssh-keygen -y -f 'Privada.pem' > 'Publica.pub'

##### Clave pública Alejandro:

ssh-rsa

```
AAAAB3NzaC1yc2EAAAADAQABAAQDOqJcEkd9jgnip/+aF/m2Ua7xVTtWCWZDFTdGo8HD6ADZxS8gq
V38qY/YuLapp5qoRySAQsKeMZDWBGRrM8ZuUp9OWfm63k8dRjMy5GEijHs6748Kj9FiaTp9/
PePthuECYQLAWLAtyCnKit+PimufaDdaJl4sK84Fz4rY8tHyKcxaAvfD7i61oWPph/
7EAnS34xTWZcF7rUvR0dL3W4MZKDr5gutSRW6GxpvLaS3m5lwetxTiGjOOjqM445ir2ZCmosXmxIKSqqD
g4GAECUm0W2g+/19KKYXp/xBpvNLKvs9oHEWllegtqBOI4IzIk0HNealmaxbDIBDpBHhCHv
```

##### Clave pública Adrián:

ssh-rsa

```
AAAAB3NzaC1yc2EAAAADAQABAAQDH3/mLPH+QScQxLdFCp+j1Zp52EFrIujq4qEw7g3luu5SOWCsx
uGpaMUN8yxSwCgEAg4BbBLU5EijhHUCV6CSQyWR8WRHMI4w5GlvbLYBNaR+uCCABfLLbTterw2/4kzK
WLm9KUZPq12O4ulALDGLh/
3eLoJGtFKwcvDJfP+zuoH1GGJrP3RdpfIT2E+ovNQGo69xgH+hUueKxOvjRMKbs0FB3CAbSdAjuMIulRnFx
owa/
HofalqOaMaoqWXeqq9Ow+im1qAmZnKAl5oz7nHJA1Njm105VutBylahXr6q4BMJMrPXuJMoyNHVYy509e
J0TYvKOvBVSPFtsnZTxowP
```

##### Clave pública Rubén:

ssh-rsa

```
AAAAB3NzaC1yc2EAAAADAQABAAQDGQVK40w8lbCmO/vWRelnC64bpYpQ5EXbyIWIGG8Xx7ir0gXlu
w6Kp3Vakm2xJp0KzpD8d4guYIBr038Vvyxe/wR69C1SKVMEEWs1MfyIuNv6J5V9kwFAHZlG/
IrJ5CajVChnKxPpeip5nO8jKUZK+zsA+6SlgFM0J9XtmTGNg0g4skqkC9PZ8f4omOBobZ0yZ1dx1Czb1HgzL
DlnyYlWi4Mdz6uBOPD4G6J/
oBEmv8Rx1CILDG3me45dT7TLLJ04g3vmVr2qPKGmPMiSeoPFKbYJU5P6uSr6okx2jEXHCF/
8Lhkxmjybh3liwGU7t3vX6PUyEGDoQqz0pfhbX6mr
```

Modificamos el archivo authorized\_keys para dejarlo de la siguiente manera:

```
ubuntu@ip-172-31-16-31:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDOqJcEkd9jgnip/+aF/m2Ua7xVTtWCWZDFTdGo8HD6
ADZxS8gqV38qY/YuLapp5qoRySAQsKeMZDWBGRrM8ZuUp9OWfm63k8dRjMy5GEijHs6748Kj9FiaTp9/
PePthuECYQLAWLAtyCnKit+PimufaDdaJl4sK84Fz4rY8tHyKcxaAvfD7i61oWPph/7EAnS34xTWZcF
7rUvR0dL3W4MZKDr5gutSRW6GxpvLaS3m5lwetxTiGjOOjqM445ir2ZCmosXmxIKSqqDg4GAECUm0W2
g+/19KKYXp/xBpvNLKvs9oHEWllegtqBOI4IzIk0HNealmaxbDIBDpBHhCHv Alejandro
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDH3/mLPH+QScQxLdFCp+j1Zp52EFrIujq4qEw7g3lu
u5SOWCsxuGpaMUN8yxSwCgEAg4BbBLU5EijhHUCV6CSQyWR8WRHMI4w5GlvbLYBNaR+uCCABfLLbTter
w2/4kzKWLm9KUZPq12O4ulALDGLh/3eLoJGtFKwcvDJfP+zuoH1GGJrP3RdpfIT2E+ovNQGo69xgH+hU
ueKxOvjRMKbs0FB3CAbSdAjuMIulRnFxAowa/HofalqOaMaoqWXeqq9Ow+im1qAmZnKAl5oz7nHJA1N
jm105VutBylahXr6q4BMJMrPXuJMoyNHVYy509eJ0TYvKOvBVSPFtsnZTxowP Adrián
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDGQVK40w8lbCmO/vWRelnC64bpYpQ5EXbyIWIGG8Xx
7ir0gXluw6Kp3Vakm2xJp0KzpD8d4guYIBr038Vvyxe/wR69C1SKVMEEWs1MfyIuNv6J5V9kwFAHZlG/
IrJ5CajVChnKxPpeip5nO8jKUZK+zsA+6SlgFM0J9XtmTGNg0g4skqkC9PZ8f4omOBobZ0yZ1dx1Czb
1HgzLDlnyYlWi4Mdz6uBOPD4G6J/oBEmv8Rx1CILDG3me45dT7TLLJ04g3vmVr2qPKGmPMiSeoPFKbY
U5P6uSr6okx2jEXHCF/8Lhkxmjybh3liwGU7t3vX6PUyEGDoQqz0pfhbX6mr Rubén
```

### 3.2. Seguridad

Hacemos una copia de seguridad de la configuración ssh por si acaso con:

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak
```

Modificamos el archivo de configuración para permitir solo conexiones con claves y no con contraseñas.

Accedemos al archivo con sudo nano /etc/ssh/sshd\_config y modificamos los atributos relevantes.

```
ubuntu@ip-172-31-16-31: ~
GNU nano 7.2          /etc/ssh/sshd_config *
PermitRootLogin no
PasswordAuthentication no
PubkeyAuthentication yes
```

## 4. Certificados para HTTPS

### 4.1. LetsEncrypt

Generamos certificados con letsEncrypt.

#### app

```
sudo mkdir -p /etc/letsencrypt/live/app.projecteGrupG1.es
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/letsencrypt/live/app.projecteGrupG1.es/privkey.pem \
-out /etc/letsencrypt/live/app.projecteGrupG1.es/fullchain.pem \
-subj "/C=ES/ST=Madrid/L=Madrid/O=ProjecteGrupG1/CN=app.projecteGrupG1.es"
```

#### backup

```
sudo mkdir -p /etc/letsencrypt/live/backup.projecteGrupG1.es
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/letsencrypt/live/backup.projecteGrupG1.es/privkey.pem \
-out /etc/letsencrypt/live/backup.projecteGrupG1.es/fullchain.pem \
-subj "/C=ES/ST=Madrid/L=Madrid/O=ProjecteGrupG1/CN=backup.projecteGrupG1.es"
```

#### test

```
sudo mkdir -p /etc/letsencrypt/live/test.projecteGrupG1.es
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/letsencrypt/live/test.projecteGrupG1.es/privkey.pem \
-out /etc/letsencrypt/live/test.projecteGrupG1.es/fullchain.pem \
-subj "/C=ES/ST=Madrid/L=Madrid/O=ProjecteGrupG1/CN=test.projecteGrupG1.es"
```

## 5. Apache

### 5.1. Virtual Hosts

app.projecteGrupG1.es

```
ubuntu@ip-172-31-16-31:~  
GNU nano 7.2 /etc/apache2/sites-available/app.projecteGrupG1.es.conf *  
<VirtualHost *:80>  
    ServerName app.projecteGrupG1.es  
    Redirect permanent / https://app.projecteGrupG1.es/  
</VirtualHost>  
  
<VirtualHost *:443>  
    ServerName app.projecteGrupG1.es  
    DocumentRoot /home/app/ftp/www  
  
    SSLEngine on  
    SSLCertificateFile /etc/letsencrypt/live/app.projecteGrupG1.es/fullchain.pem  
    SSLCertificateKeyFile /etc/letsencrypt/live/app.projecteGrupG1.es/privkey.pem  
  
    ErrorLog /home/app/logs/error.log  
    CustomLog /home/app/logs/access.log combined  
  
    ProxyPreserveHost On  
    ProxyPass / http://127.0.0.1:8000/  
    ProxyPassReverse / http://127.0.0.1:8000/  
  
    RequestHeader set X-Forwarded-Proto "https"  
    RequestHeader set X-Forwarded-Port "443"  
    RequestHeader set X-Real-IP "%{REMOTE_ADDR}s"  
    RequestHeader set X-Forwarded-For "%{REMOTE_ADDR}s"  
</VirtualHost>
```

backup.projecteGrupG1.es

```
ubuntu@ip-172-31-16-31:~  
GNU nano 7.2 /etc/apache2/sites-available/backup.projecteGrupG1.es.conf *  
<VirtualHost *:80>  
    ServerName backup.projecteGrupG1.es  
    Redirect permanent / https://backup.projecteGrupG1.es/  
</VirtualHost>  
  
<VirtualHost *:443>  
    ServerName backup.projecteGrupG1.es  
    DocumentRoot /home/backup/ftp/fitxers  
  
    SSLEngine on  
    SSLCertificateFile /etc/letsencrypt/live/backup.projecteGrupG1.es/fullchain.pem  
    SSLCertificateKeyFile /etc/letsencrypt/live/backup.projecteGrupG1.es/privkey.pem  
  
    <Directory /home/backup/ftp/fitxers>  
        Options Indexes FollowSymLinks  
        AuthType Basic  
        AuthName "Acceso Restringido - Backups"  
        AuthUserFile /etc/apache2/.htpasswd_backup  
        Require valid-user  
    </Directory>  
  
    ErrorLog /var/log/apache2/backup_error.log  
    CustomLog /var/log/apache2/backup_access.log combined  
</VirtualHost>
```

Además del VirtualHost para backup, creamos credenciales para acceder por contraseña con  
sudo htpasswd -c /etc/apache2/.htpasswd\_backup app y le ponemos 12345  
Para loguearnos: User → app; Password → 12345

test.projecteGrupG1.es

```
ubuntu@ip-172-31-16-31:~  
GNU nano 7.2 /etc/apache2/sites-available/test.projecteGrupG1.es.conf *  
<VirtualHost *:80>  
    ServerName test.projecteGrupG1.es  
    Redirect permanent / https://test.projecteGrupG1.es/  
</VirtualHost>  
  
<VirtualHost *:443>  
    ServerName test.projecteGrupG1.es  
    DocumentRoot /home/app/ftp/test  
  
    SSLEngine on  
    SSLCertificateFile /etc/letsencrypt/live/test.projecteGrupG1.es/fullchain.pem  
    SSLCertificateKeyFile /etc/letsencrypt/live/test.projecteGrupG1.es/privkey.pem  
  
    ProxyPreserveHost On  
    # El entorno test correrá en el puerto 8001  
    ProxyPass / http://127.0.0.1:8001/  
    ProxyPassReverse / http://127.0.0.1:8001/  
  
    RequestHeader set X-Forwarded-Proto "https"  
    RequestHeader set X-Forwarded-Port "443"  
  
    ErrorLog /var/log/apache2/test_error.log  
    CustomLog /var/log/apache2/test_access.log combined  
</VirtualHost>
```

Hecho esto, configuraremos el archivo hosts para que las ip apunten a nuestros nombres.

```
batoi@AlejandroRico: ~/Documentos/Segundo/PI/despliegue  
GNU nano 7.2 /etc/hosts *  
127.0.0.1      localhost  
127.0.1.1      batoi-SSD-2024  
  
# Pagina PI  
3.213.82.57    app.projecteGrupG1.es  
3.213.82.57    backup.projecteGrupG1.es  
3.213.82.57    test.projecteGrupG1.es
```

## 6. Servidor FTP

### 6.1. vsftpd.conf

Instalamos vsftpd y configuramos el archivo vsftpd.conf para que cumpla con los requisitos de puertos, la ip de nuestra máquina y que podamos permitir el uso a ciertos usuarios.

```
ubuntu@ip-172-31-16-31:~  
GNU nano 7.2          /etc/vsftpd.conf  
# ======  
# CONFIGURACIÓN VSFTPD - ProjecteGrupG1  
# ======  
  
# === Acceso Básico ===  
anonymous_enable=NO  
local_enable=YES  
write_enable=YES  
local_umask=022  
  
# === Encarcelar usuarios (Chroot) ===  
chroot_local_user=YES  
allow_writeable_chroot=YES  
  
# === Modo Pasivo (Requisito 3.3: puertos 30000-30050) ===  
pasv_enable=YES  
pasv_min_port=30000  
pasv_max_port=30050  
pasv_address=3.213.82.57  
  
# === Seguridad ===  
force_local_logins_ssl=NO  
force_local_data_ssl=NO  
ssl_enable=NO  
pam_service_name=vsftpd  
secure_chroot_dir=/var/run/vsftpd/empty  
  
# === Logs ===  
xferlog_enable=YES  
xferlog_file=/var/log/vsftpd.log  
xferlog_std_format=YES  
log_ftp_protocol=YES  
  
# === Restringir usuarios a lista específica ===  
userlist_enable=YES  
userlist_file=/etc/vsftpd.userlist  
userlist_deny=YES  
  
listen=YES  
listen_ipv6=NO
```

## 6.2. vsftpd.userlist

Permitimos que el usuario ubuntu pueda utilizar el servicio ftp

```
ubuntu@ip-172-31-16-31:~$ sudo cat /etc/vsftpd.userlist
ubuntu
```

Creamos contraseñas para vsftpd para los usuarios app y backup

sudo passwd app → app1234  
sudo passwd backup → backup1234

## 6.3. Comprobación vsftpd

Vemos el estado del servicio vsftpd para comprobar que esta active (running) y nos conectamos con app desde filezilla para comprobar que funcione correctamente.

```
ubuntu@ip-172-31-16-31:~$ sudo systemctl status vsftpd
● vsftpd.service - vsftpd FTP server
  Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; preset: en>
  Active: active (running) since Mon 2026-02-23 14:39:18 UTC; 9s ago
    Main PID: 3905 (vsftpd)
      Tasks: 1 (limit: 4665)
     Memory: 712.0K (peak: 1.5M)
        CPU: 8ms
       CGroup: /system.slice/vsftpd.service
                 └─3905 /usr/sbin/vsftpd /etc/vsftpd.conf

Feb 23 14:39:18 ip-172-31-16-31 systemd[1]: Starting vsftpd.service - vsftpd FTP>
Feb 23 14:39:18 ip-172-31-16-31 systemd[1]: Started vsftpd.service - vsftpd FTP>
```

The screenshot shows the FileZilla interface with two panes. The left pane shows the local directory structure at `/home/batoi/`, which includes standard system directories like .config, bin, logs, boot, and cdrom. The right pane shows the remote directory structure at `/logs`, which contains subdirectories `ftp`, `test`, `www`, and `logs`. Under `logs`, there are two files: `access.log` and `error.log`. Both files have a size of approximately 1.3 KB and were modified on 23/02/26 at 15:38. The permissions for both files are `-rw-r--r--` and the owner is `root`.

Nombre de archivo	Tamaño de	Tipo de arch	Última modifica	Permisos	Propietario/C
..					
.CloudOfflineMirrors	Directorio		26/09/25 17:10...		
.cache	Directorio		23/02/26 15:46...		
.config	Directorio		23/02/26 15:46...		
.docker	Directorio		04/02/26 18:42...		
.dotnet	Directorio		16/09/25 09:18...		
.Fonts	Directorio		16/09/25 09:38...		
.ganttpoint.d	Directorio		26/09/25 17:00...		
19 archivos y 35 directorios. Tamaño total: 2,3 MB					
2 archivos. Tamaño total: 4,7 KB					

## 7. Despliegue de la aplicación

### 7.1. APP

Para subir los archivos a la máquina virtual, lo haremos desde git en vez de el ftp que acabamos de configurar para que sea mas sencillo poder actualizar el proyecto y luego, por ftp enviaremos los archivos .env necesarios con las credenciales, ya que estos están dentro den .gitignore y no se copiarán.

Hacemos un git clone de nuestro repositorio en app/ftp/www de lo que sería nuestro proyecto completo y vemos los siguientes archivos:

Nombre de archivo	Tamaño de	Tipo de arch	Última modifica	Permisos	Propietario/C
package-lock.json	186,1 KB	json-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
package.json	725 B	json-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
phpunit.xml	1,2 KB	xml-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
postcss.config.js	93 B	js-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
readme.md	0 B	md-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
tailwind.config.js	541 B	js-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001
vite.config.js	364 B	js-archivo	23/02/26 16:...	-rw-rw-r--	1001 1001

18 archivos y 15 directorios. Tamaño total: 646,8 KB

Ahora movemos a www los archivos .env y .env.app que sería los necesarios para el despliegue en producción de nuestra aplicación. Esto lo haremos desde filezilla.

Seleccionamos los dos archivos que nos interesan y los arrastramos al directorio de nuestra máquina virtual.

Nombre de archivo	Tamaño de	Tipo de archivo	Última modificación
venv		Directorio	09/01/26 16:44:...
.editorconfig	252 B	Archivo	23/12/25 16:13:...
.env	700 B	Archivo	18/02/26 16:30:...
.env.app	1,6 KB	app-archivo	23/02/26 15:59:...
.env.example	400 B	example-archivo	03/11/25 19:11:...
.env.test	1,6 KB	test-archivo	23/02/26 16:01:...
.env.testing	513 B	testing-archivo	18/02/26 16:30:...
.gitattributes	186 B	Archivo	23/12/25 16:13:...

2 archivos seleccionados. Tamaño total: 2,3 KB

```
ubuntu@ip-172-31-16-31:/$ sudo ls -la /home/app/ftp/www/
total 776
drwxr-xr-x 19 app app    4096 Feb 23 15:36 .
drwxr-xr-x  4 app app    4096 Feb 19 18:14 ..
-rw-rw-r--  1 app app     252 Feb 23 15:27 .editorconfig
-rw-rw-r--  1 app app    730 Feb 23 15:36 .env
-rw-r--r--  1 app app   1596 Feb 23 15:36 .env.app
```

Ahora que todo parece estar listo, hacemos docker compose up -d para construir y levantar los contenedores pro primera vez. Sorprendentemente no ha fallado por ahora y tenemos lo siguiente:

```
ubuntu@ip-172-31-16-31:/home/app/ftp/www
=> [stage-0 4/6] RUN usermod -u 1000 www-data && groupmod -g 1000 www-da 0.3s
=> [stage-0 5/6] RUN mkdir -p /var/www/html /home/www-data/.npm &&      c 0.2s
=> [stage-0 6/6] WORKDIR /var/www/html                                0.1s
=> exporting to image                                                 10.1s
[+] up 91/91ting layers                                              8.4s
✓ Image redis:7-alpine          Pulled                           5.7ss
✓ Image n8nio/n8n:latest       Pulled                           64.0s
✓ Image pi-laravel-app        Built                            204.2s
✓ Image phpmyadmin:latest      Pulled                           38.8s
✓ Image node:20                Pulled                           49.1s
✓ Image nginx:1.27-alpine      Pulled                           5.5ss
✓ Image mysql:8.4              Pulled                           42.8s
✓ Network www_default          Created                          0.0s
✓ Volume www_n8n_data          Created                          0.0s
✓ Volume www_db_data           Created                          0.0s
✓ Container pi-laravel-redis  Created                          0.3s
✓ Container pi-laravel-n8n    Created                          0.3s
✓ Container pi-laravel-db     Created                          0.3s
✓ Container pi-laravel-app    Created                          0.1s
✓ Container pi-laravel-phpmyadmin Created                      0.1s
✓ Container pi-laravel-nginx  Created                          0.1s
✓ Container pi-laravel-vite   Created                          0.1s
```

Ahora tenemos que instalar las dependencias necesarias de nuestro proyecto para que todo funcione correctamente.

Añadimos en el .gitignore en local el archivo para evitar pisarlo en un futuro igual que con los .env.

Debido a que en local tenemos contenedores en docker con nginx y en remoto servimos la aplicación a través de apache y que tenemos en el proyecto mezcla de laravel para backend (+ vistas blade para frontend iniciales que ya no se utilizan) y vue para el frontend, debemos configurar alguna ruta de routes/web.php y routes/api.php junto con nginx/default.conf para que las rutas no entren en bucles cílicos y funcionen correctamente.

```
ubuntu@ip-172-31-16-31:/home/app/ftp/www$ cat nginx/default.conf
server {
    listen 80;
    server_name _;

    root /var/www/html/public;
    index index.php index.html;

    # API → Laravel (PRIMERO)
    location ^~ /api {
        try_files $uri $uri/ /index.php?$query_string;
    }

    # PHP-FPM
    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_pass app:9000;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param DOCUMENT_ROOT $document_root;
    }

    # Assets (muy importante antes que SPA fallback)
    location /assets/ {
        try_files $uri =404;
        expires 7d;
        access_log off;
    }

    # SPA fallback SOLO para lo que no sea API ni archivo real
    location / {
        try_files $uri $uri/ /index.html;
    }

    location ~ /\. {
        deny all;
    }
}
```

## Arquitectura de la aplicación en producción:

Aquí se muestra un diagrama general del despliegue en el que podemos ver el flujo de la información desde que el usuario accede a la aplicación hasta que se obtiene una respuesta desde la base de datos.

El cliente se conecta a través del protocolo HTTPS al servidor Apache, que actúa como reverse proxy y se encarga de gestionar el certificado SSL.

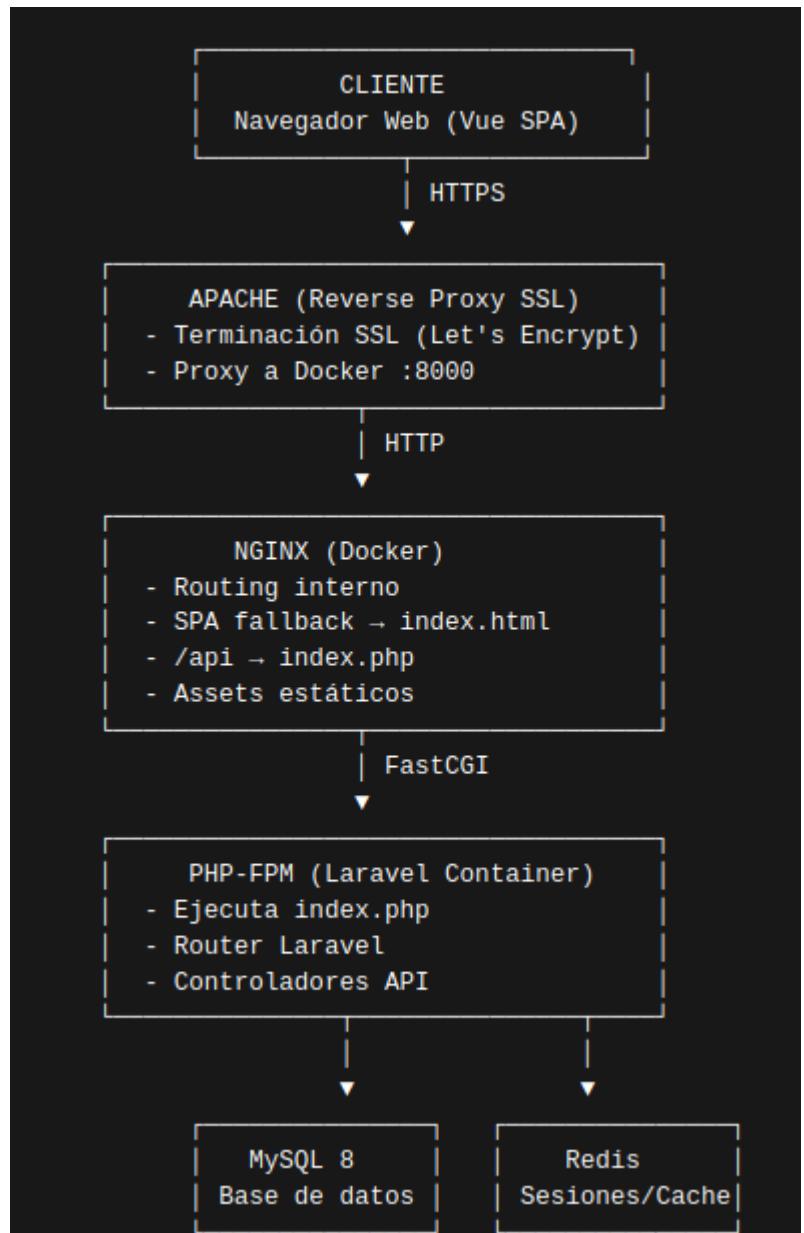
El servidor Apache redirige las peticiones al servidor Nginx, que se encuentra ejecutándose dentro de un contenedor Docker.

Nginx analiza el tipo de petición recibida:

Si es una ruta de la API (/api/\*), la redirige a PHP-FPM, donde se ejecuta Laravel.

Si es una ruta de la aplicación web (como /login o /register), devuelve el archivo index.html para que Vue gestione la navegación.

Si es un recurso estático (como archivos JavaScript o CSS), lo sirve directamente.



Cuando la petición se dirige a Laravel, este procesa la lógica de negocio y se comunica con MySQL para obtener o almacenar datos, y con Redis para gestionar sesiones o caché.

Finalmente, la respuesta vuelve siguiendo el mismo recorrido hasta llegar al cliente.

## Arquitectura Lógica SPA + API:

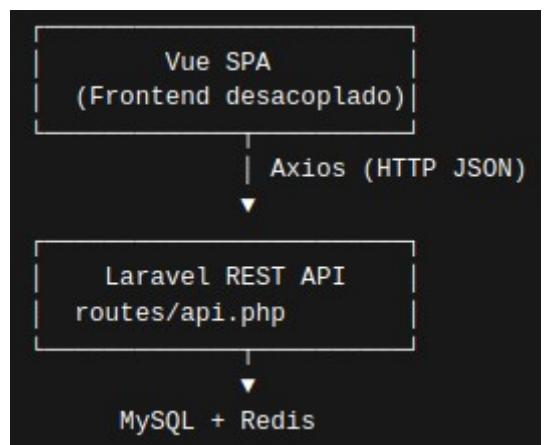
Aquí se muestra un diagrama lógico del funcionamiento interno de la aplicación, donde se aprecia la separación entre frontend y backend.

El cliente carga una aplicación desarrollada en Vue, que funciona como una Single Page Application (SPA). Cuando el usuario interactúa con la aplicación, Vue realiza peticiones HTTP mediante Axios a la API REST desarrollada en Laravel.

Laravel recibe estas peticiones a través de sus rutas definidas en routes/api.php, ejecuta la lógica correspondiente en los controladores y consulta la base de datos MySQL cuando es necesario.

Una vez procesada la información, Laravel devuelve una respuesta en formato JSON, que Vue interpreta y muestra dinámicamente en la interfaz sin necesidad de recargar la página completa.

Este modelo permite separar claramente la parte visual (frontend) de la lógica de negocio y acceso a datos (backend), facilitando el mantenimiento y la escalabilidad del sistema.



A continuación se muestran algunas de las vistas de la aplicación web para demostrar que todo funciona correctamente y no es solo la vista principal.

PI-Laravel Vue App — Mozilla Firefox

Lanzamiento del Laboratorio Instances | EC2 | us-east-1 PI-Laravel Vue App +

app.projectegrupg1.es

BK BITKEYS Buscar...

Mewgenics  
Ya disponible

Últimas novedades

-5%

-10%

The screenshot shows a web browser window with the title 'PI-Laravel Vue App — Mozilla Firefox'. The address bar displays 'app.projectegrupg1.es'. The page content features a large, semi-transparent image of a cartoon cat's face. Overlaid on this image is the word 'Mewgenics' in large white letters and the text 'Ya disponible' below it. At the bottom of the page, there is a section titled 'Últimas novedades' (Last news) with two visible promotional banners. The first banner on the left has a '-5%' discount badge. The second banner on the right has a '-10%' discount badge. The browser interface includes a tab bar with 'Lanzamiento del Laboratorio', 'Instances | EC2 | us-east-1', and 'PI-Laravel Vue App'. The toolbar at the top has various icons for file operations, search, and navigation. The address bar also contains a list of frequently visited sites.

PI-Laravel Vue App — Mozilla Firefox

Lanzamiento del Laboratorio × Instances | EC2 | us-east-1 × PI-Laravel Vue App × +

app.projectegrupg1.es/products/3

Windows FCT WF Aules horario DWEC DWES DIW DAW IPE NuB SOS DIGI PI Chat Git Codec Udemy Angular AWS api ext 8000 8080 n8n video ProjecteGrupG1

BK BITKEYS Buscar... 🔎



## Baldur's Gate III

40.00€ **36.00€**

Sumérgete en un RPG épico ambientado en el universo de Dungeons & Dragons. Baldur's Gate III ofrece una narrativa rica, decisiones que influyen directamente en la historia y un sistema de combate por turnos profundo y estratégico.

Plataforma:

PlayStation 5 Steam Xbox Series X

Añadir al carrito 🛒 ⚡

### Comentarios

**Adrián** ★ 4

Tenetur quia doloremque et molestiae non et totam fuga perspiciatis nihil enim.

PI-Laravel Vue App — Mozilla Firefox

Lanzamiento del Laboratorio Instances | EC2 | us-east-1 PI-Laravel Vue App +

app.projectegrupg1.es/checkout

BK BITKEYS Buscar... 🔎

Finalizar Compra

Resumen del Pedido

Celeste 1 x 10.00 € 10.00 €  
Plataforma: Steam

Total: 10.00 €

Método de Pago

Tarjeta guardada  Nueva tarjeta

Selecciona tu tarjeta  
\*\*\*\*\* 1113 - ADMIN USER

Volver al carrito Pagar 10.00 €

This screenshot shows a checkout page for a purchase of the game 'Celeste' on Steam. The total cost is 10.00 €. The payment method is set to a saved card, and the card number is partially visible as '\*\*\*\*\* 1113 - ADMIN USER'. There are buttons for 'Volver al carrito' (Return to Cart) and 'Pagar 10.00 €' (Pay 10.00 €).

## 7.2. Test

Para desplegar los tests, podemos realizar una copia del despliegue realizado para producción en app y modificar el archivo .env para que coincida con el .env.test que se ha generado anteriormente. De esta manera podemos aprovechar todos los ajustes realizados para que funcione bien la web y tener así un entorno real para realizar pruebas con la aplicación desplegada en un entorno real. Esto es interesante debido a que tenemos añadidas opciones en nuestra aplicación para poder modificar de forma manual productos.

Debido a que no tenemos demasiado espacio en la máquina virtual de AWS, no vamos a duplicar los archivos, pero en nuestro caso, al querer una instancia igual a la de producción pero con una base de datos distinta para poder hacer las pruebas dejando solo este .env y el resto igual debería ser suficiente:

```
# =====
# APP - Configuración de Test
# =====
APP_NAME=ProjecteGrupG1
APP_ENV=testing
APP_KEY=base64:Y776kFynd6PFMSIWQwTzpB7nc8x9JteYYcy2E2rIc5w=
APP_DEBUG=false
APP_URL=https://test.projecteGrupG1.es

LOG_CHANNEL=stack
LOG_LEVEL=error

# =====
# BASE DE DATOS - Test
# =====
DB_CONNECTION=mysql
DB_HOST=db
DB_PORT=3306
DB_DATABASE=pi_laravel_test
DB_USERNAME=pi
DB_PASSWORD=pi

# =====
# CACHE / COLAS / SESIONES - Test
# =====
BROADCAST_DRIVER=log
CACHE_DRIVER=redis
FILESYSTEM_DISK=local
QUEUE_CONNECTION=redis
SESSION_DRIVER=redis
SESSION_LIFETIME=120

REDIS_HOST=redis
REDIS_PASSWORD=null
REDIS_PORT=6379
```

```

# =====
# GOOGLE AUTH - Test
# =====
GOOGLE_CLIENT_ID=782388634943-
00eldr0npg8i37ndph980g76m4844oku.apps.googleusercontent.com
GOOGLE_CLIENT_SECRET=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
GOOGLE_REDIRECT_URI=https://test.projecteGrupG1.es/api/auth/google/callback

# =====
# FRONTEND (Vue/Vite) - Test
# =====
FRONTEND_URL=https://test.projecteGrupG1.es

# =====
# PROXY / HEADERS - Importante detrás de Apache
# =====
TRUSTED_PROXIES=*

```

Hemos cambiado estas líneas:

APP\_ENV=testing ← para tener el entorno de test

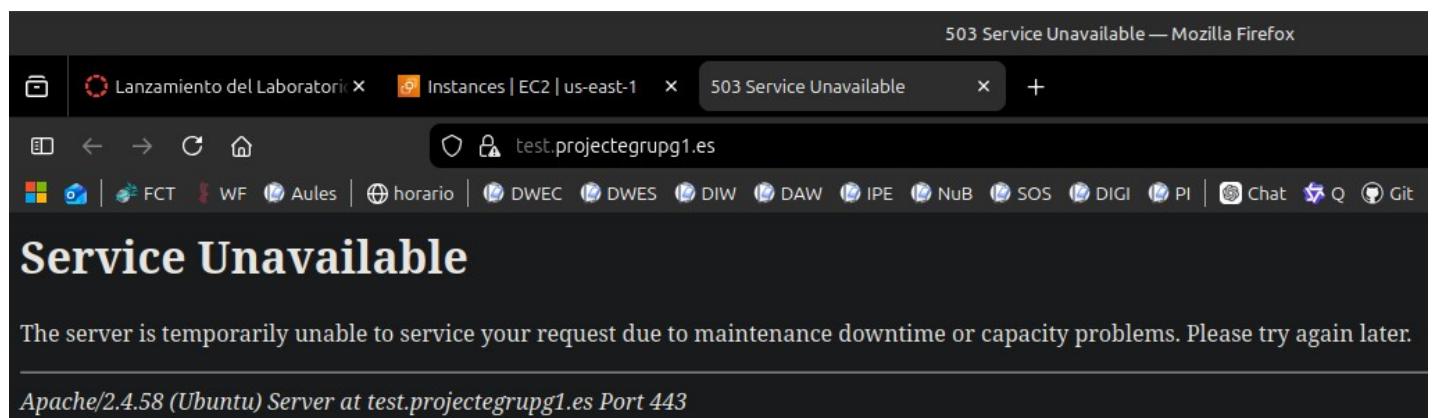
APP\_URL=<https://test.projecteGrupG1.es> ← la dirección de acceso a la web

DB\_DATABASE=pi\_laravel\_test ← para tener una nueva base de datos (habría que modificar los controladores para redirigirlos a esa nueva base de datos de test)

GOOGLE\_REDIRECT\_URI=<https://test.projecteGrupG1.es/api/auth/google/callback> ← para poder seguir logueando desde google en el entorno de test

FRONTEND\_URL=<https://test.projecteGrupG1.es> ← para que vue pueda servir el frontend en test.

Ahora mismo el entorno está vacío y sin levantar docker ni nada, pero esta vista nos dice que debería estar todo configurado para poder funcionar correctamente.



### 7.3. Backup

El entorno de backup está ya preparado para poder almacenar ahí los datos que consideremos oportunos. Si entramos desde la web a <https://backup.projectegrupg1.es/> vemos que nos pide las credenciales como se ha configurado anteriormente para poder acceder al sitio en el que tendremos almacenadas las copias de seguridad.

