

Práctica 0 - Instalación nginx y primer espacio web básico

Mi primera aplicación web

1.-Objetivos de la práctica.....	2
2.- Recursos utilizados.....	2
3.- Procedimiento.....	2
3.1 Instalación de VirtualBox.....	3
3.2 Creación de la máquina virtual para la Instalación Ubuntu-server.....	4
3.3 Instalación del sistema operativo Ubuntu-server.....	5
3.3.1 Conexión al servidor a través de SSH.....	5
4. Instalación de NGINX.....	6
4.1 Instalación de modulos de PHP.....	6
4.2 Instalación de Mysql Server.....	7
5. Configuración del servidor NGINX.....	7
5.1 Configuración de puerto NGINX.....	7
5.2 Añadiendo modulos PHP a NGINX.....	8
5.3 Añadiendo autenticación básica a NGINX.....	8
5.4 Configuración del directorio root de NGINX.....	8
5.5 Configuración de lectura de archivos de NGINX.....	8
5.6 Reinicio del servicio NGINX.....	8
5.7 Solucionando conflicto de servidores web.....	9
6. Creación del repositorio de nuestra aplicación.....	9
6.1 Creación del archivo de prueba index.php.....	9
6.2 Configuración de los parametros del repositorio.....	10
6.3 Guardamos los cambios en nuestro repositorio.....	10
7. Aplicación Extagram.....	10
7.1 Archivos de la aplicación.....	11
8. Configuración de la base de datos.....	14
8.1 Creación de la base de datos en Mysql.....	15
8.2 Creación de la tabla POSTS en Extagram_db.....	15
8.3 Creación de usuario en MYSQL.....	16
8.4 Test de conexión a la base de datos con el usuario creado.....	17
9. Protección básica de nuestra web.....	17
10. Test de nuestra aplicación extagram.com.....	19
11. Conclusions.....	20

1.-Objetivos de la práctica

La práctica 1 consiste en el despliegamiento del servidor web con nociones sobre Nginx preprocesando vía módulos y bases de datos.

2.- Recursos utilizados

Para realizar esta práctica se han utilizado los recursos siguientes:

- **VirtualBox** para crear una maquina virtual con el sistema operativo Ubuntu-Server.
- **Nginx** para ofrecer el servicio web en nuestro servidor.,
- **Modulos de php**: “php php-fpm php-mysql php-dev” para poder compilar codigo php en el servidor.
- **Mysql-server** para poder ofrecer el servicio de bases de datos.
- **ssh-server/openssh-server** para poder conectarnos a nuestro servidor Ubuntu desde una máquina externa y gestionarlo de una manera más sencilla.

3.- Procedimiento

Instalación de Virtualbox para poder crear máquinas virtuales.

El objetivo en esta práctica es hacer la instalación de un servidor web, en este caso utilizaremos el sistema operativo: Ubuntu-server.

3.1 Instalación de VirtualBox

Para la instalar virtualbox se ha de editar el archivo: /etc/apt/sources.list

Y añadir la siguiente linea:

```
deb [arch=amd64 signed-by=/usr/share/keyrings/oracle-virtualbox-2016.gpg]  
https://download.virtualbox.org/virtualbox/debian bullseye contrib
```

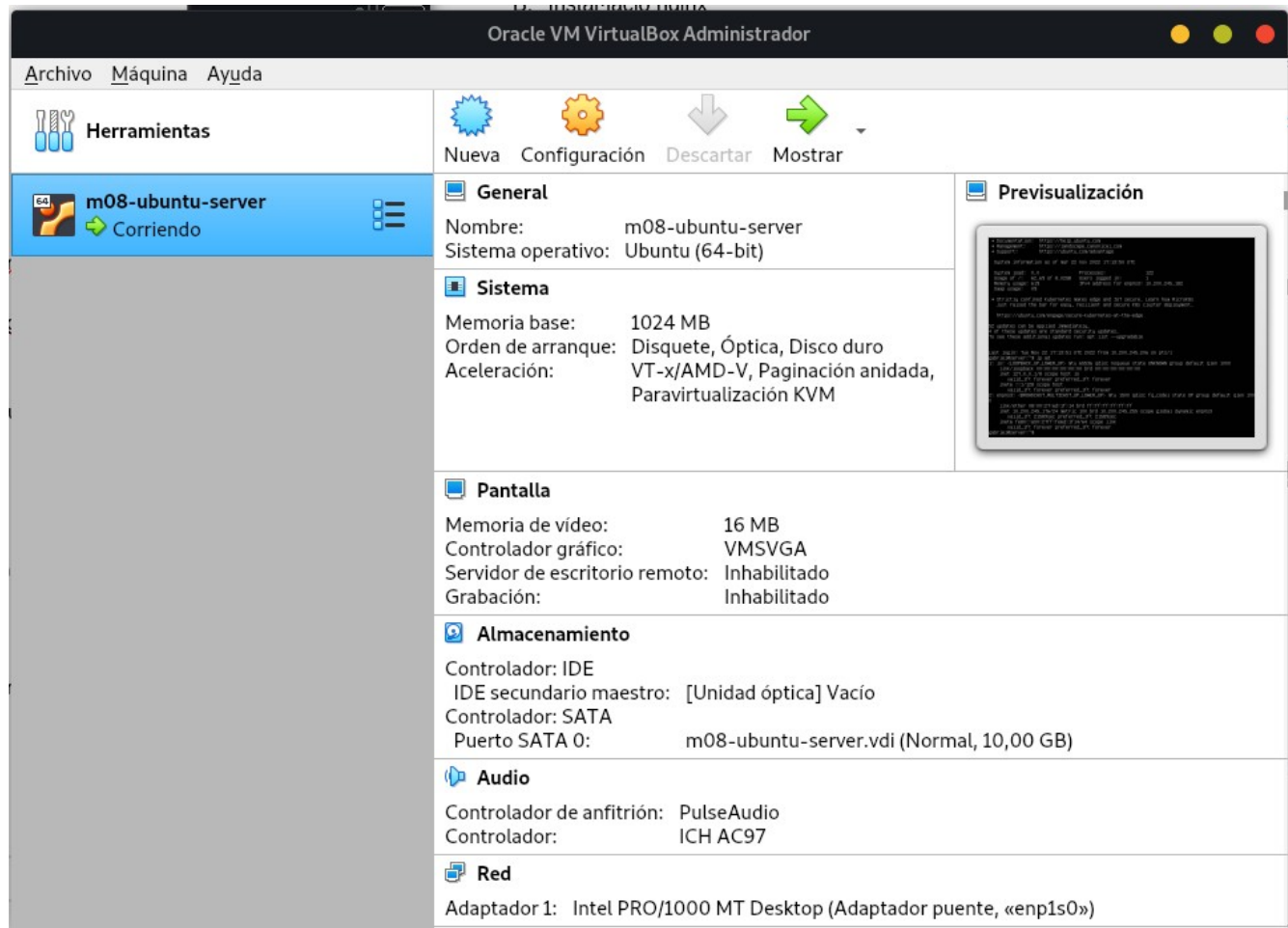
```
a222603gc@g06:~$ cat /etc/apt/sources.list  
# deb http://ftp.caliu.cat/debian/ bullseye main  
  
deb http://ftp.caliu.cat/debian/ bullseye main contrib non-free  
deb-src http://ftp.caliu.cat/debian/ bullseye main  
  
deb http://security.debian.org/debian-security bullseye-security maincontrib non-free  
deb-src http://security.debian.org/debian-security bullseye-security main  
  
# bullseye-updates, to get updates before a point release is made;  
# see https://www.debian.org/doc/manuals/debian-reference/ch02.en.html#_updates_and_backports  
deb http://ftp.caliu.cat/debian/ bullseye-updates main contrib non-free  
deb-src http://ftp.caliu.cat/debian/ bullseye-updates main  
  
deb [arch=amd64 signed-by=/usr/share/keyrings/oracle-virtualbox-2016.gpg] https://download.virtual  
box.org/virtualbox/debian bullseye contrib
```

Una vez realizado este punto hay que actualizar los repositorios e instalar Virtualbox

```
sudo apt-get update  
sudo apt-get install virtualbox-6.1
```

3.2 Creación de la máquina virtual para la Instalación Ubuntu-server.

Se ha procedido a la creación de una máquina virtual con los siguientes parámetros:



Importante la configuración de la red en adaptador puente para poder tener conexión con la maquina anfitrión tanto para la conexión de ssh como para poder hacer peticiones web.

Una vez creada la máquina virtual se “monta” la .ISO de ubuntu server para realizar su instalación.

En este caso hemos descargado la versión Ubuntu Server 22.04 LTS. Una versión que tiene el soporte extendido.

<https://ubuntu.com/download/server>

3.3 Instalación del sistema operativo Ubuntu-server.

Una vez que hemos creado la máquina virtual y hemos insertado la ISO de instalación de Ubuntu server realizamos su instalación con los siguientes parametros:

- Teclado español
- Red DHCP
- Seleccionada la opción de instalar SSH (para poder conectarnos posteriormente por ssh)
- El formateo de disco lo dejamos por defecto ya que no necesitamos particiones especiales.



3.3.1 Conexión al servidor a través de SSH

Para conectarnos al servidor ejecutamos la instrucción:

```
a222603gc@g06:~$ ssh gabriel@10.200.245.176
gabriel@10.200.245.176's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-53-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of mié 23 nov 2022 18:35:15 UTC

System load:  0.703125      Processes:           118
Usage of /:   62.8% of 8.02GB Users logged in:        0
Memory usage: 59%          IPv4 address for enp0s3: 10.200.245.176
Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

48 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Wed Nov 23 18:35:16 2022
gabriel@server:~$
```

4. Instalación de NGINX

Realizamos la instalación de NGINX para poder ofrecer el servicio web en nuestro servidor.

Para su instalación ejecutamos el siguiente comando:

```
sudo apt-get install nginx
```

Comprobamos su instalación con el comando:

```
sudo service nginx status
```

```
gabriel@server:~$ sudo service nginx status
[sudo] password for gabriel:
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: en
   Active: active (running) since Wed 2022-11-23 18:34:53 UTC; 7min ago
     Docs: man:nginx(8)
   Process: 646 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_proces
   Process: 713 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (co
  Main PID: 724 (nginx)
    Tasks: 2 (limit: 1030)
   Memory: 7.7M
      CPU: 30ms
   CGroup: /system.slice/nginx.service
           └─724 "nginx: master process /usr/sbin/nginx -g daemon on; master
             └─727 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" ""
nov 23 18:34:53 server systemd[1]: Starting A high performance web server and a
nov 23 18:34:53 server systemd[1]: Started A high performance web server and a
línes 1-16/16 (END)
```

4.1 Instalación de modulos de PHP

Instalamos los modulos necesarios para compilar codigo php en nuestro servidor nginx.

Ejecutamos los comandos:

```
sudo apt install apache2-utils
sudo apt install -y php php-fpm php-mysql php-dev
```

4.2 Instalación de Mysql Server

Instalamos el servidor de bases de datos Mysql, para la correcta función de nuestra aplicación ya que necesita consultar y guardar datos.

```
sudo apt-get install mysql-server
```

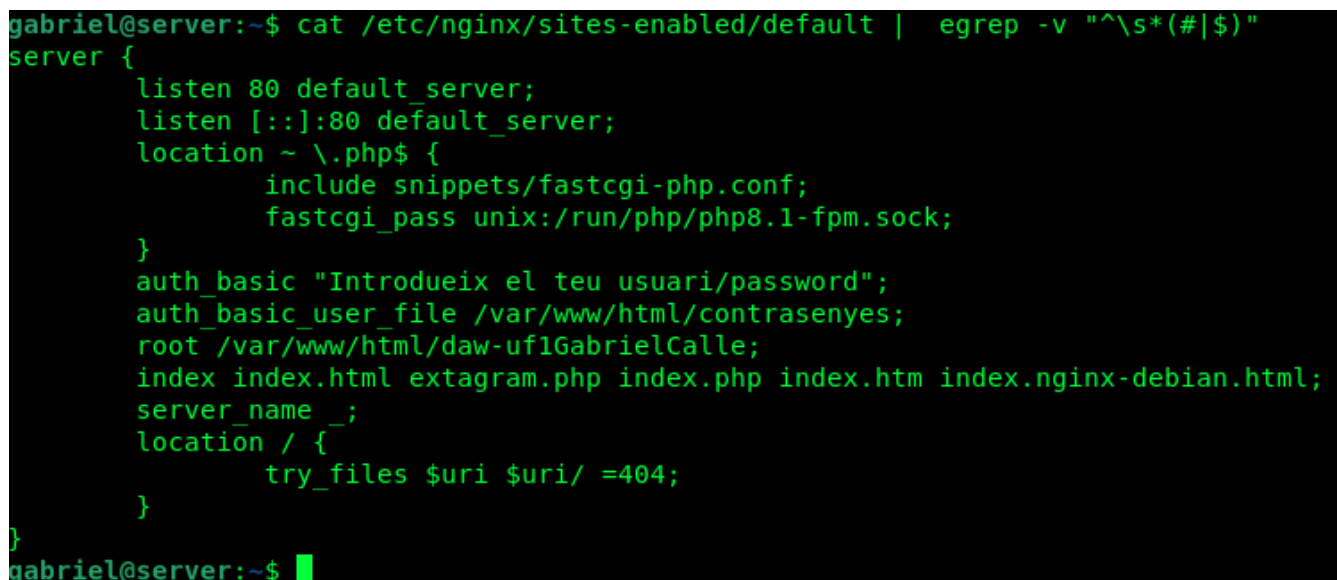
5. Configuración del servidor NGINX

Para configurar nuestro servidor web NGINX necesitamos editar el fichero /etc/nginx/sites-enabled/default

Donde añadiremos las lineas necesarias para que nuestro servidor Nginx pueda procesar código PHP.

El fichero debe quedar como en la imagen.

```
cat /etc/nginx/sites-enabled/default | egrep -v "^\s*(#|$)"
```



```
gabriel@server:~$ cat /etc/nginx/sites-enabled/default | egrep -v "^\s*(#|$)"
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    location ~ /\.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
    }
    auth_basic "Introdueix el teu usuari/password";
    auth_basic_user_file /var/www/html/contrasenyes;
    root /var/www/html/daw-uf1GabrielCalle;
    index index.html extagram.php index.php index.htm index.nginx-debian.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

5.1 Configuración de puerto NGINX

Se indica el puerto de escucha de nuestro servidor web:

```
listen 80 default_server;
listen [::]:80 default_server;
```

5.2 Añadiendo modulos PHP a NGINX

Se indica que va a compilar PHP con las siguientes lineas.

```
location ~ \.php$ {  
    include snippets/fastcgi-php.conf;  
    fastcgi_pass unix:/run/php/php8.1-fpm.sock;  
}
```

5.3 Añadiendo autenticación básica a NGINX

Se indica que se requiere autenticación con las siguientes lineas

```
auth_basic "Introdueix el teu usuari/password";  
auth_basic_user_file /var/www/html/contrasenyes;
```

5.4 Configuración del directorio root de NGINX

Se indica el directorio donde alojamos nuestra aplicación. Es el directorio raíz que se mostrara cuando alguien haga una petición web:

```
root /var/www/html/daw-uf1GabrielCalle;
```

5.5 Configuración de lectura de archivos de NGINX

Se indica la prioridad y los ficheros que enseñara cuando alguien acceda a la raíz del proyecto:

```
index index.html extagram.php index.php index.htm index.nginx-debian.html;
```

5.6 Reinicio del servicio NGINX

Una vez realiza toda la configuración del servidor NGINX reiniciamos el servicio para poder cargar la nueva configuración.

```
sudo service nginx restart
```


5.7 Solucionando conflicto de servidores web

Al instalar los modulos para compilar codigo PHP si se instala de manera colateral el servidor Apache, hemos de cambiar el puerto de escucha del servidor Apache.

Para ello hemos de modificar el siguiente archivo /etc/apache2/sites-enabled/000-default.conf y cambiar el puerto de escucha. En este caso hemos asignado el 1010.

```
gabriel@server:~$ sudo cat /etc/apache2/sites-enabled/000-default.conf | egrep -v "^#\s*(#|$)"
<VirtualHost *:1010>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
gabriel@server:~$
```

Una vez realiza la configuración reiniciamos el servicio:

```
sudo service apache2 restart
```

6. Creación del repositorio de nuestra aplicación

Creamos un nuevo repositorio en nuestro perfil de GITLAB el cual se llamará:

```
https://gitlab.inf-edt.org/a222603gc/daw-uf1GabrielCalle
```

6.1 Creación del archivo de prueba index.php

Creamos un archivo index.php que contendrá el siguiente código:

```
<?php echo "<h1>Hello World</h1>";?>
```

Lo abrimos en el navegador y si no hay ningún problema se podrá visualizar: "Hello world" lo cual indica que se está procesando PHP correctamente.

Realizamos una petición a nuestro servidor <http://10.200.245.176/index.php>

6.2 Configuración de los parametros del repositorio

Una vez que confirmamos que nuestro servidor está procesando correctamente los archivos php podemos continuar con la práctica. Por lo que configuración los siguientes parametros de git:

```
git config --global user.name "Gabriel Calle"  
git config --global user.email "gabriel.calle92@gmail.com"
```

6.3 Guardamos los cambios en nuestro repositorio

Inicializamos nuestro repositorio y guardamos los cambios con los siguientes comandos:

```
git init  
git add .  
git commit -m "primera revisión"  
git remote add origin https://gitlab.inf-edt.org/a222603gc/m8\_uf1.git  
git push origin main
```

7. Aplicación Extagram

Una vez realizada toda la configuración y pruebas anteriores. Procedemos a la creación de nuestra página extagram. Para lo cual vamos a crear los siguientes archivos en nuestro servidor web.

Nos ubicamos en el directorio de nuestro proyecto:

```
cd /var/www/html/daw-uf1GabrielCalle/
```

Creamos los siguientes archivos:

- extagram.php
- preview.svg
- style.css
- upload.php
- uploads/ -> (Directorio donde se subirán las imagenes de la app)

7.1 Archivos de la aplicación

Contenido de los archivos de la aplicación:

Extagram.php

```
<link rel="stylesheet" href="style.css">

<form method="POST" enctype="multipart/form-data" action="upload.php">
  <input type="text" name="post" placeholder="Escribe algo...">
  <input id="file" type="file" name="photo"
onchange="document.getElementById('preview').src=window.URL.createObjectURL(event.target.files[0])">
  <label for="file">
    
  </label>
  <input type="submit">
</form>
<?php

$db = new mysqli("db.extagram.edt", "extagram_admin", "pass123", "extagram_db");

foreach ($db->query("SELECT * FROM posts") as $fila) {
  echo "<div class='post'>";
  echo "<p style='text-align: center;'>".$fila['post'].</p>";
  if (!empty($fila['photourl'])) {
    echo "<img style='margin: 0 auto;' width='200px' src='uploads/".
$fila['photourl']."'>";
  }
  echo "</div>";
}
?>
```

Style.css

```
body {
  background: #fafafa;
  font-family: sans;
  margin: 0;
}

form {
  display: flex;
```

```
flex-direction: column;
justify-content: center;
align-items: center;
gap: 1em;
background: white;
border-bottom: 1px solid #dbdbdb;
padding: 8px;
}

input[type=text] {
border: 1px solid #dbdbdb;
padding: 8px;
width: 300px;
}
input[type=submit] {
background: #0096f7;
color: white;
border: 0;
border-radius: 3px;
width: 300px;
padding: 8px;
}
#file { display: none; }
#preview { max-width: 300px; }

.post {
max-width: 600px;
margin: 0 auto;
background: white;
display: flex;
flex-direction: column;
border: 1px solid #dbdbdb;
border-radius: 3px;
margin-bottom: 24px;
}
.post p { padding: 16px; }
```

Upload.php

```
<?php

if (!empty($_POST["post"])) {

    $photoid;
    if(!empty($_FILES['photo']['name'])) {
        $photoid = uniqid();
        echo $photoid;
    }
}
```

```
        move_uploaded_file($_FILES['photo']['tmp_name'], 'uploads/' . $photoid);
    }
    $db = new mysqli("db.extagram.edt", "extagram_admin", "pass123", "extagram_db");
    $stmt = $db->prepare("INSERT INTO posts(post,photourl) VALUES (?,?)");
    $stmt->bind_param("ss", $_POST["post"], $photoid);
    $stmt->execute();
    $stmt->close();
}
header("location: /");
?>
```

Preview.svg

```
<?xml version="1.0" encoding="UTF-8"?>
<svg version="1.1" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100"
width="300" height="300">
<g>
<rect width="100" height="100" fill="#cecece"/>
<path fill="ffffff" transform="translate(25 25)" d="M48.1,26.3c0,4.3,0,7.2-0.1,8.8c-
0.2,3.9-1.3,6.9-3.5,9s-5.1,3.3-9,3.5c-1.6,0.1-4.6,0.1-8.8,0.1c-4.3,0-7.2,0-8.8-0.1 c-3.9-
0.2-6.9-1.3-9-3.5c-2.1-2.1-3.3-5.1-3.5-9c-0.1-1.6-0.1-4.6-0.1-8.8s0-7.2,0.1-8.8c0.2-
3.9,1.3-6.9,3.5-9 c2.1-2.1,5.1-3.3,9-3.5c1.6-0.1,4.6-0.1,8.8-
0.1c4.3,0,7.2,0,8.8,0.1c3.9,0.2,6.9,1.3,9,3.5s3.3,5.1,3.5,9 C48,19.1,48.1,22,48.1,26.3z
M28.8,8.7c-1.3,0-2,0-2.1,0c-0.1,0-0.8,0-2.1,0c-1.3,0-2.3,0-2.9,0c-0.7,0-1.6,0-2.7,0.1 c-
1.1,0-2.1,0.1-2.9,0.3c-0.8,0.1-1.5,0.3-2,0.5c-0.9,0.4-1.7,0.9-2.5,1.6c-0.7,0.7-1.2,1.5-
1.6,2.5c-0.2,0.5-0.4,1.2-0.5,2 s-0.2,1.7-0.3,2.9c0,1.1-0.1,2-
0.1,2.7c0,0.7,0.1,7,0.2,9c0,1.3,0.2,0.2,1s0,0.8,0.2,1c0,1.3,0.2,3,0.2,9c0,0.7,0.1,6,0.1,2.7
c0,1.1,0.1,2.1,0.3,2.9s0.3,1.5,0.5,2c0.4,0.9,0.9,1.7,1.6,2.5c0.7,0.7,1.5,1.2,2.5,1.6c0.5,0.2
,1.2,0.4,2,0.5
c0.8,0.1,1.7,0.2,2.9,0.3s2,0.1,2.7,0.1c0.7,0.1,7,0.2,9c0.1,3,0.2,0.2,1,0c0.1,0,0.8,0.2,1,0c
1.3,0.2,3,0.2,9,0 c0.7,0.1,6,0.2,7-0.1c1.1,0.2,1-0.1,2.9-0.3c0.8-0.1,1.5-0.3,2-0.5c0.9-
0.4,1.7-0.9,2.5-1.6c0.7-0.7,1.2-1.5,1.6-2.5 c0.2-0.5,0.4-1.2,0.5-2c0.1-0.8,0.2-1.7,0.3-
2.9c0-1.1,0.1-2,0.1-2.7c0-0.7,0-1.7,0-2.9c0-1.3,0-2,0-2.1s0-0.8,0-2.1 c0-1.3,0-2.3,0-
2.9c0-0.7,0-1.6-0.1-2.7c0-1.1-0.1-2.1-0.3-2.9c-0.1-0.8-0.3-1.5-0.5-2c-0.4-0.9-0.9-1.7-1.6-
2.5 c-0.7-0.7-1.5-1.2-2.5-1.6c-0.5-0.2-1.2-0.4-2-0.5c-0.8-0.1-1.7-0.2-2.9-0.3c-1.1,0-2-
0.1-2.7-0.1C31.1,8.7,30.1,8.7,28.8,8.7z M34.4,18.5c2.1,2.1,3.2,4.7,3.2,7.8s-1.1,5.6-
3.2,7.8c-2.1,2.1-4.7,3.2-7.8,3.2c-3.1,0-5.6-1.1-7.8-3.2c-2.1-2.1-3.2-4.7-3.2-7.8 s1.1-
5.6,3.2-7.8c2.1-2.1,4.7-3.2,7.8-3.2C29.7,15.3,32.3,16.3,34.4,18.5z M31.7,31.3c1.4-
1.4,2.1-3.1,2.1-5s-0.7-3.7-2.1-5.1 c-1.4-1.4-3.1-2.1-5.1-2.1c-2,0-3.7,0.7-5.1,2.1s-2.1,3.1-
2.1,5.1s0.7,3.7,2.1,5c1.4,1.4,3.1,2.1,5.1,2.1 C28.6,33.4,30.3,32.7,31.7,31.3z
M39.9,13c0.5,0.5,0.8,1.1,0.8,1.8c0,0.7-0.3,1.3-0.8,1.8c-0.5,0.5-1.1,0.8-1.8,0.8 s-1.3-0.3-
1.8-0.8c-0.5-0.5-0.8-1.1-0.8-1.8c0-0.7,0.3-1.3,0.8-1.8c0.5-0.5,1.1-0.8,1.8-
0.8S39.4,12.5,39.9,13z"/>
</g>
</svg>
```

8. Configuración de la base de datos

Para configurar nuestra base de datos hemos de asignar una contraseña a nuestro usuario root de la base de datos. Para ello seguimos las siguientes instrucciones:

- `service mysql stop`
- `sudo mkdir /var/run/mysqld`
- `sudo chown mysql /var/run/mysqld`
- `mysqld_safe --skip-grant-tables &`
- `sudo mysql --user=root mysql`
- `UPDATE mysql.user SET authentication_string=null WHERE User='root';`
- `flush privileges;`
- `ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'contrasenya';`
- `flush privileges;`
- `exit`
- `sudo killall -u mysql`
- `service mysql start`
- `mysql -uroot -p`

Comprobamos que los cambios han tenido efecto y que el servicio está corriendo correctamente:

```
gabriel@server:/var/www/html/daw-uf16GabrielCalle$ sudo service mysql status
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2022-11-23 18:34:57 UTC; 55min ago
     Process: 644 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
    Main PID: 750 (mysqld)
      Status: "Server is operational"
      Tasks: 38 (limit: 1030)
     Memory: 365.1M
        CPU: 35.897s
    CGroup: /system.slice/mysql.service
            └─750 /usr/sbin/mysqld

nov 23 18:34:53 server systemd[1]: Starting MySQL Community Server...
nov 23 18:34:57 server systemd[1]: Started MySQL Community Server.
```

8.1 Creación de la base de datos en Mysql

Vamos a crear la base de datos de nuestra aplicación con el siguiente comando:

```
CREATE DATABASE extragram_db
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| extragram_db |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
```

8.2 Creación de la tabla POSTS en Extragram_db

Creemos la tabla necesaria para que funcione nuestra aplicación con el siguiente comando:

```
CREATE TABLE posts (
  id int PRIMARY KEY NOT NULL AUTO_INCREMENT,
  post varchar(255),
  photourl varchar(255)
);
```

```
mysql> show tables;
+-----+
| Tables_in_extragram_db |
+-----+
| posts |
+-----+
1 row in set (0,01 sec)
```

```
mysql> describe posts;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id | int | NO | PRI | NULL | auto_increment |
| post | varchar(255) | YES | | NULL | |
| photourl | varchar(255) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0,02 sec)
```

8.3 Creación de usuario en MYSQL

Para nuestra práctica vamos a crear el correspondiente usuario que nuestra aplicación necesita. En este caso vamos a crear el usuario: extagram_admin.

Para ello ejecutamos la siguiente instrucción:

```
CREATE USER 'extagram_admin'@'localhost' IDENTIFIED BY 'pass123';
```

Asignamos los correspondientes permisos al usuario y actualizamos:

```
GRANT ALL PRIVILEGES ON extagram_db.* TO "extagram_admin"@"localhost" ;  
  
FLUSH PRIVILEGES;
```

Confirmamos que se han guardado correctamente los permisos y creación de usuario:

```
mysql> SHOW GRANTS FOR extagram_admin@localhost;  
+-----+  
| Grants for extagram_admin@localhost |  
+-----+  
| GRANT USAGE ON *.* TO `extagram_admin`@`localhost` |  
| GRANT ALL PRIVILEGES ON `extagram_db`.* TO `extagram_admin`@`localhost` |  
+-----+  
2 rows in set (0,00 sec)
```

```
mysql> SELECT user,host FROM mysql.user;  
+-----+  
| user          | host      |  
+-----+  
| debian-sys-maint | localhost |  
| extagram_admin  | localhost |  
| mysql.infoschema | localhost |  
| mysql.session   | localhost |  
| mysql.sys       | localhost |  
| root            | localhost |  
+-----+  
6 rows in set (0,00 sec)
```


8.4 Test de conexión a la base de datos con el usuario creado.

Realizamos un test de conexión con el usuario que acabamos de crear con el siguiente comando:

```
mysql -h localhost -u extagram_admin --database extagram_db -p
```

```
gabriel@server:/var/www/html/daw-uf1GabrielCalle$ mysql -h localhost -u extagram_admin --database extagram_db -p
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select * from posts;
+-----+-----+-----+
| id | post      | photourl |
+-----+-----+-----+
| 1  | dasdasd   | 637d13cad26e1 |
| 2  | facebook  | 637d1527e349f |
| 3  | apuntes   | 637d1558cdc3c |
| 4  | consulta mysql | 637e5cde0744e |
+-----+-----+-----+
4 rows in set (0,00 sec)

mysql>
```

9. Protección básica de nuestra web

Para proteger nuestra página web vamos a crear un archivo de contraseñas en el un directorio al cual nuestro servidor web pueda tener acceso, en este caso lo vamos a crear en el directorio: `/var/www/html/` el archivo se llamará "contrasenyas". Para crear el archivos vamos a ejecutar la siguiente instrucción:

```
htpasswd -c /var/www/html/contrasenyas gabrielCalle
```

```
gabriel@server:~$ sudo htpasswd -c /var/www/html/contrasenyes gabrielCalle
[sudo] password for gabriel:
New password:
Re-type new password:
Adding password for user gabrielCalle
gabriel@server:~$ sudo cat /var/www/html/contrasenyes
gabrielCalle:$apr1$YNIItMNTC$LufaqUN5PILpi94s02lXL/
gabriel@server:~$
```

Una vez ejecutada la instrucción podemos ver que nos ha solicitado una contraseña que se almacenará de forma segura. Si miramos el archivo creado vemos que la contraseña está encriptada:

```
gabriel@server:/var/www/html/daw-uf1GabrielCalle$ cat /var/www/html/contrasenyes
gabrielCalle:$apr1$YNIItMNTC$LufaqUN5PILpi94s02lXL/
```

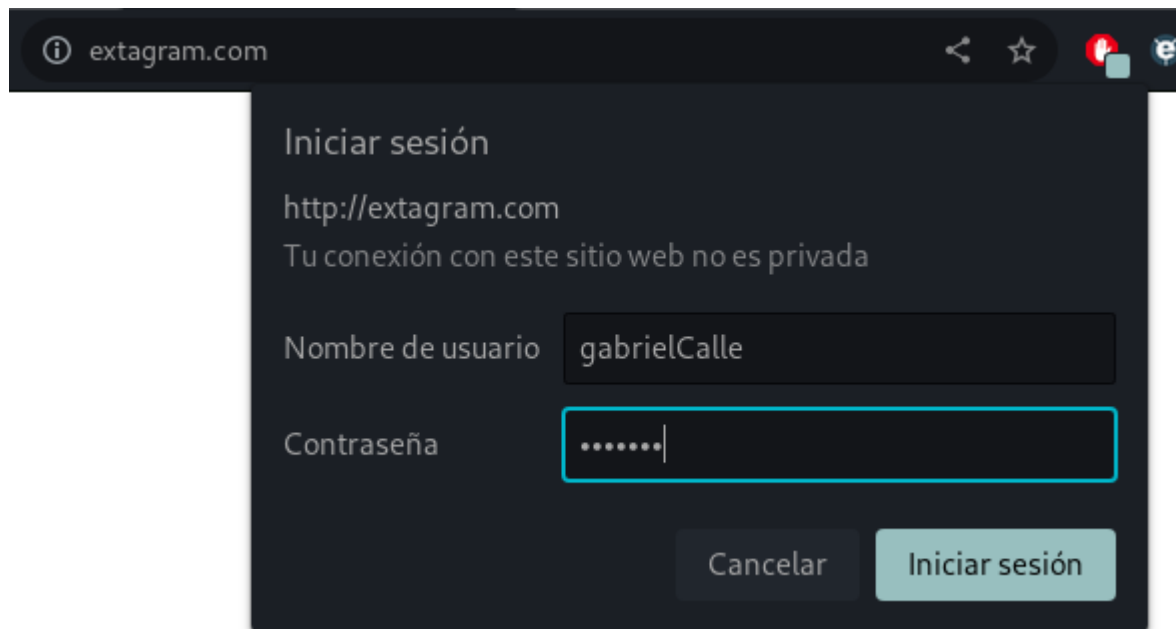
La protección básica de nuestra aplicación funcionará porque anteriormente ya hemos editado nuestro archivo de configuración de nginx /etc/nginx/sites-enabled/default en el cual hemos añadido las líneas:

```
auth_basic "Introdueix el teu usuari/password";
auth_basic_user_file /var/www/html/contrasenyes;
```

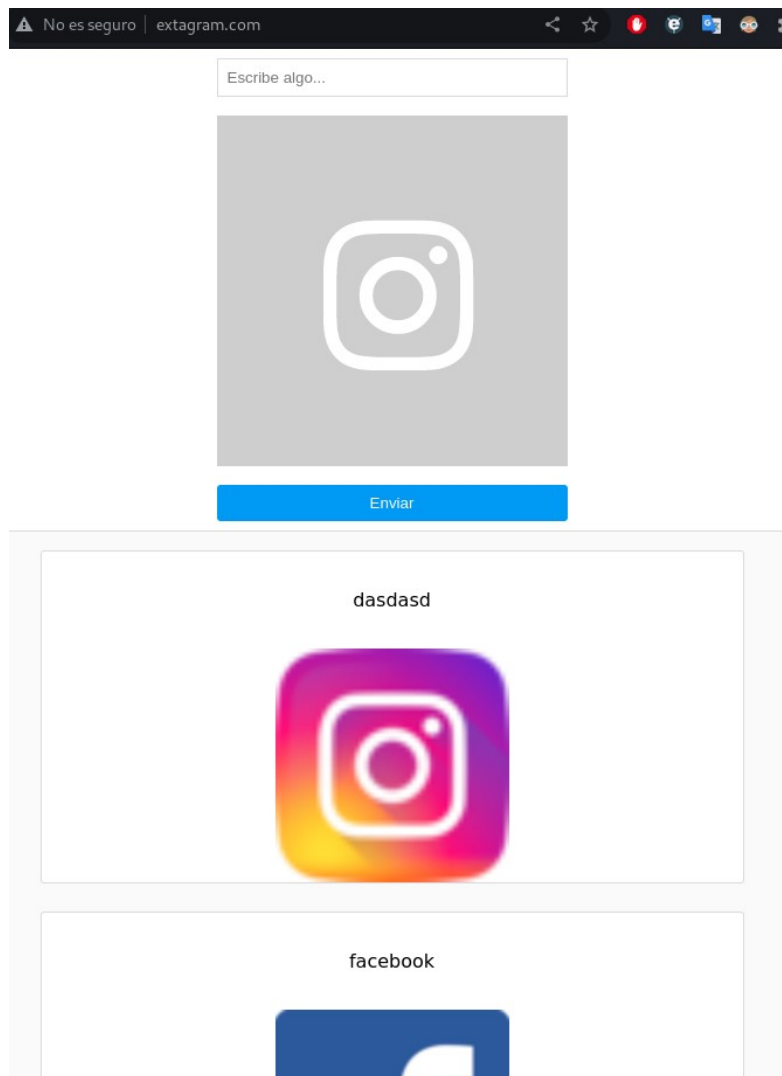
```
gabriel@server:~$ cat /etc/nginx/sites-enabled/default | egrep -v "^\s*(#|$)"
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    location ~ /\.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
    }
    auth_basic "Introdueix el teu usuari/password";
    auth_basic_user_file /var/www/html/contrasenyes;
    root /var/www/html/daw-uf1GabrielCalle;
    index index.html extagram.php index.php index.htm index.nginx-debian.html;
    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

10. Test de nuestra aplicación extagram.com

Para confirmar que hemos realizado correctamente toda la configuración e instalación de los recursos necesarios para iniciar nuestra aplicación, realizamos petición web a nuestro servidor y vamos a publicar fotografías.



Como podemos ver la autenticación básica está funcionando correctamente. Nos pide usuario y contraseña los cuales ya hemos definido anteriormente.



11. Conclusions

Hemos podido ser capaces de instalar y configurar un servidor web que procesa código PHP, instalar y configurar un servidor de bases de datos, conectarnos por remoto a nuestro servidor a través de SSH, configurar una aplicación que pone en práctica la correcta configuración de todos los servicios y establecer una protección básica de acceso en nuestras peticiones web.