



UNIVERSIDAD DE GRANADA

SWAP

Práctica 1

Trabajo individual

Nuria Manzano Mata

Profesor: José Manuel Soto Hidalgo



Tiempo dedicado: 13,5h

ÍNDICE

1. Introducción.....	2
2. Desarrollo tareas básicas.....	2
2.1. B1. Configuración del entorno.....	2
2.2. B2. Creación del Dockerfile.....	2
2.3. B3. Uso de Docker Compose.....	3
2.4. B4. Despliegue y verificación de Contenedores.....	5
2.5. B5. Pruebas Básicas.....	13
2.6. Análisis propuesta IA.....	14
3. Desarrollo tareas avanzadas.....	15
3.1. A1: Personalización del Dockerfile.....	15
3.2. A2. Creación de contenedores con otros servidores web.....	19
3.3. A3. Gestión Avanzada de Redes.....	28
3.4. A4. Automatización con Scripts.....	30
3.5. A5. Monitoreo y Logging.....	33
3.6. Análisis propuesta IA.....	36
Bibliografía:.....	37

1. Introducción

Ésta práctica busca introducir al uso de Docker mediante el despliegue de servidores web utilizando Apache y PHP. El **desarrollo de este documento se estructura de manera progresiva**, abordando cada tarea con un **doble propósito**: explicar detalladamente los procedimientos y soluciones implementadas, y justificar el razonamiento detrás de cada decisión. Para facilitar la comprensión, se incluyen capturas de pantalla que **muestran el proceso**. Debido a la extensión del documento, algunos apartados se presentan de manera concisa, mostrando los resultados más significativos o los aspectos más importantes.

El .zip entregado contiene dos carpetas, una llamada “P1” donde está todo el desarrollo de las tareas básicas y otra carpeta llamada “P1-ADVANCED” donde está todo el desarrollo de las tareas avanzadas.

2. Desarrollo tareas básicas

2.1. B1. Configuración del entorno

- Crear un directorio en tu máquina local llamado web_usuarioUGR.

```
[ 2025-03-20 12:41:37 ] Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
[○ → ls
DockerfileApache_mmnuria docker-compose.yml
web_mmnuria]
```

- Dentro de este directorio, crear un archivo index.php que muestre “SWAP-nombre del usuario” y la dirección IP del servidor Apache”.

```
[ 2025-03-20 12:45:46 ] Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
[○ → cd web_mmnuria/
[ 2025-03-20 12:45:48 ] Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1/web_mmnuria
index.php

[ 2025-03-20 12:45:53 ] Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1/web_mmnuria
[○ → cat index.php
<!DOCTYPE html>
<html>
<head>
    <title>Práctica 1 SWAP - Nuria Manzano Mata</title>
</head>
<body>
    <h1>Práctica 1 SWAP - Nuria Manzano Mata</h1>
    <?php
        $server_ip = $_SERVER['SERVER_ADDR'];
        echo "<p>La dirección IP del servidor es: " . $server_ip . "</p>";
    ?>
</body>
</html>
```

2.2. B2. Creación del Dockerfile

- Crear un archivo Dockerfile en la raíz del proyecto llamado DockerfileApache_usuarioUGR.

```
2025-03-20 12:46:49 ✉ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
[○ → ls
DockerfileApache_mmnuria docker-compose.yml web_mmnuria
```

- Usar una imagen base de Linux, instalar Apache, PHP y herramientas de red para comprobar conectividad entre máquinas.

```
2025-03-20 12:46:51 ✉ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
[○ → cat DockerfileApache_mmnuria
#versión 11 de debian
FROM debian:bullseye-slim
# Instalación Apache, PHP y herramientas de red
RUN apt-get update && apt-get install -y \
    apache2 \
    php \
    libapache2-mod-php \
    iputils-ping \
    net-tools \
    curl \
    iproute2 \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

# Habilita el módulo Apache para PHP (necesario para que Apache procese archivos PHP)
RUN a2enmod php7.4

# Establecer directorio de trabajo
WORKDIR /var/www/html

# Puerto HTTP estándar
EXPOSE 80

# Mantener apache en ejecución
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

2.3. B3. Uso de Docker Compose

Crear un archivo docker-compose.yml con las siguientes características:

- Crear una imagen llamada usuarioUGR-apache-image:p1 a partir del Dockerfile DockerfileApache_usuarioUGR.

```
2025-03-20 12:48:05 ✉ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
[○ → cat docker-compose.yml
version: '3'

services:
  web1:
    build:
      context: .
      dockerfile: DockerfileApache_mmnuria
      image: mgnuria-apache-image:p1
    container_name: web1
    volumes:
      - ./web_mmnuria:/var/www/html
    networks:
      red_web:
        ipv4_address: 192.168.10.2
      red_servicios:
        ipv4_address: 192.168.20.2
    ports:
      - "8080:80"
    restart: unless-stopped
```

- Crear 8 contenedores llamados webX, donde X es un número de 1 a 8 con volúmenes donde se monte el directorio “web_usuarioUGR” en el directorio raíz de Apache en el contenedor.

```

web1:
build:
  context: .
  dockerfile: DockerfileApache_mmnuria
image: mmnuria-apache-image:p1
container_name: web1
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.2
  red_servicios:
    ipv4_address: 192.168.20.2
ports:
  - "8080:80"
restart: unless-stopped

web2:
image: mmnuria-apache-image:p1
container_name: web2
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.3
  red_servicios:
    ipv4_address: 192.168.20.3
ports:
  - "8081:80"
restart: unless-stopped

web3:
image: mmnuria-apache-image:p1
container_name: web3
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.4
  red_servicios:
    ipv4_address: 192.168.20.4
ports:
  - "8082:80"
restart: unless-stopped

web4:
image: mmnuria-apache-image:p1
container_name: web4
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.5
  red_servicios:
    ipv4_address: 192.168.20.5
ports:
  - "8083:80"
restart: unless-stopped

web5:
image: mmnuria-apache-image:p1
container_name: web5
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.6
  red_servicios:
    ipv4_address: 192.168.20.6
ports:
  - "8084:80"
restart: unless-stopped

web6:
image: mmnuria-apache-image:p1
container_name: web6
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.7
  red_servicios:
    ipv4_address: 192.168.20.7
ports:
  - "8085:80"
restart: unless-stopped

web7:
image: mmnuria-apache-image:p1
container_name: web7
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.8
  red_servicios:
    ipv4_address: 192.168.20.8
ports:
  - "8086:80"
restart: unless-stopped

web8:
image: mmnuria-apache-image:p1
container_name: web8
volumes:
  - ./web_mmnuria:/var/www/html
networks:
  red_web:
    ipv4_address: 192.168.10.9
  red_servicios:
    ipv4_address: 192.168.20.9
ports:
  - "8087:80"
restart: unless-stopped

```

- Añadir las dos redes al contenedor, una red llamada red_web con dirección 192.168.10.0/24 y otra red llamada red_servicios con dirección 192.168.20.0/24.

```

networks:
  red_web:
    driver: bridge
    ipam:
      config:
        - subnet: 192.168.10.0/24
  red_servicios:
    driver: bridge
    ipam:
      config:
        - subnet: 192.168.20.0/24

```

2.4. B4. Despliegue y verificación de Contenedores

- Lanzar los contenedores con docker-compose up

```

2025-03-20 12:07:57 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
o ~ docker-compose up -d
[+] Running 10/10
✓ Network p1_red_web     Created          0.1s
✓ Network p1_red_servicios Created          0.1s
✓ Container web8          Started         0.5s
✓ Container web3          Started         0.4s
✓ Container web5          Started         0.4s
✓ Container web2          Started         0.4s
✓ Container web6          Started         0.4s
✓ Container web4          Started         0.4s
✓ Container web7          Started         0.4s
✓ Container web1          Started         0.4s

```

- Utilizar docker ps para verificar que todos los contenedores estén en ejecución.

2025-03-20 12:30:38 ☺ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1						
● ○ → docker-compose ps	NAME	IMAGE	COMMAND	SERVICE	CREATED	STATUS
PORTS						
web1	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web1	8 seconds ago	Up 5 seconds	0.0.0.0:8080->80/tcp
web2	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web2	8 seconds ago	Up 5 seconds	0.0.0.0:8081->80/tcp
web3	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web3	8 seconds ago	Up 5 seconds	0.0.0.0:8082->80/tcp
web4	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web4	8 seconds ago	Up 5 seconds	0.0.0.0:8083->80/tcp
web5	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web5	8 seconds ago	Up 5 seconds	0.0.0.0:8084->80/tcp
web6	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web6	8 seconds ago	Up 5 seconds	0.0.0.0:8085->80/tcp
web7	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web7	8 seconds ago	Up 5 seconds	0.0.0.0:8086->80/tcp
web8	mmnuria-apache-image:p1	"apache2ctl -D FOREG..."	web8	8 seconds ago	Up 5 seconds	0.0.0.0:8087->80/tcp

- Comprobar que cada contenedor tiene una IP asignada en las redes red_web y red_servicios.

```
2025-03-20 13:03:17 ☺ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
● ○ → docker network inspect p1_red_web
[
  {
    "Name": "p1_red_web",
    "Id": "744f4fd9ddb725f9790b5e96eaf276326c311a29b416b71fcb6165fb06b1313",
    "Created": "2025-03-20T11:30:35.783358482Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "192.168.10.0/24"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "588c841d34fc75e811fd4044923be2149c1cfb232e314b7696d4408a5fad416e": {
        "Name": "web5",
        "EndpointID": "1a6da5e0951b845205a20e658cad395f18fa0a44eae3b7f23ca9600f040f276e",
        "MacAddress": "02:42:c0:a8:0a:06",
        "IPv4Address": "192.168.10.6/24",
        "IPv6Address": ""
      },
      "8b3c5e00507146327e98a79f96d24e592f70d83f6ddacad0dfbac3c2495a449a": {
        "Name": "web6",
        "EndpointID": "2a6da5e0951b845205a20e658cad395f18fa0a44eae3b7f23ca9600f040f276e",
        "MacAddress": "02:42:c0:a8:0a:07",
        "IPv4Address": "192.168.10.7/24",
        "IPv6Address": ""
      }
    }
  }
]
```

```

"8b3c5e00507146327e98a79f96d24e592f70d83f6ddacad0dfbac3c2495a449a": {
    "Name": "web3",
    "EndpointID": "fa398c71dc03d6fb4dd4a2889e67655f46da4216176673e1f22d4e8708957a2",
    "MacAddress": "02:42:c0:a8:0a:04",
    "IPv4Address": "192.168.10.4/24",
    "IPv6Address": ""
},
"8c52e58d1251e935ff9517abbea399915b0b7007dcebdc61aa7477cc0479d76d": {
    "Name": "web6",
    "EndpointID": "c53bf26603ce01eb86cbeccc8777825351c86f08d8373ab60b38fa90ceb6e097",
    "MacAddress": "02:42:c0:a8:0a:07",
    "IPv4Address": "192.168.10.7/24",
    "IPv6Address": ""
},
"b3f193d9a159b2df8d80650605475157259cd73dd1da8466e9105c477c846c02": {
    "Name": "web4",
    "EndpointID": "3e1bac626ecc8c90beb471741aaade9e244a1a06afc5a1ab6e7a669331cbdbb",
    "MacAddress": "02:42:c0:a8:0a:05",
    "IPv4Address": "192.168.10.5/24",
    "IPv6Address": ""
},
"d16b341acc10c8f53d87a9c078b0b12db1b9f0a4c46281613d86479d3fd16e12": {
    "Name": "web8",
    "EndpointID": "5e02e380bf01d6db3244d9c2d4ec6e1698ee6af24dfc219ec14862d1d37cf7d1",
    "MacAddress": "02:42:c0:a8:0a:09",
    "IPv4Address": "192.168.10.9/24",
    "IPv6Address": ""
},
"d52718ecab6a69fdee060d7d42f56329bcb891274fe54b65cca740f082e25e68": {
    "Name": "web1",
    "EndpointID": "a0aa6a9ca6e8fdf15ebd6dbe8be02f68d771006af7d6a8ca02de03d4e744375e",
    "MacAddress": "02:42:c0:a8:0a:02",
    "IPv4Address": "192.168.10.2/24",
    "IPv6Address": ""
},
"f7e9d1b92c64d863bfd63c989c5029a7f49176ff3f47f886ffc3cf0c85f6219c": {
    "Name": "web2",
    "EndpointID": "31c1768bcb02fdab35a96d142d6f8f8bf266dc80d0763327fe2f90eb58ee99f6",
    "MacAddress": "02:42:c0:a8:0a:03",
    "IPv4Address": "192.168.10.3/24",
    "IPv6Address": ""
},
"fa52943db9ad2b1d418ae02dfcd58f270a3c65a3b4829e8827a0931710e83c03": {
    "Name": "web7",
    "EndpointID": "c39b6d5c3c238b9b5328ec53b5b72b5704bdfa935d879931b49346d411aac43",
    "MacAddress": "02:42:c0:a8:0a:08",
    "IPv4Address": "192.168.10.8/24",
    "IPv6Address": ""
}
},
"Options": {},
"Labels": {
    "com.docker.compose.network": "red_web",
    "com.docker.compose.project": "p1",
    "com.docker.compose.version": "2.23.3"
}

```

```
2025-03-24 17:42:43 ⓘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/
SWAP/P1
● o → docker network inspect p1_red_servicios
[
{
    "Name": "p1_red_servicios",
    "Id": "b17e7288897a814cd79f58df22c8ed7615b4f7afe679e1fb55878d30683d035f",
    "Created": "2025-03-20T11:30:35.857138892Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
        "Driver": "default",
        "Options": null,
        "Config": [
            {
                "Subnet": "192.168.20.0/24",
                "Gateway": "192.168.20.1"
            }
        ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
        "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
        "0a2d8cba86dad9ab27bf22c0fbcb872e666969e280a015156875cb02f3ac84c5": {
            "Name": "web7",
            "EndpointID": "46ed5b7b136c6517c14307a2ceb152b02b5aba877196f3bd8aa1
8d47b538ba6d",
            "MacAddress": "02:42:c0:a8:14:08",
            "IPv4Address": "192.168.20.8/24",
            "IPv6Address": ""
        }
    }
},
```

```
        "1310efe52023c76186dda702f14f96dfc00ab2f2d753b78f53ddaa7759f9180d": {
            "Name": "web3",
            "EndpointID": "7bc014bd0d4da6a86959508b3e4072e881171db0d17d6389e751
bca67c233cbf",
            "MacAddress": "02:42:c0:a8:14:04",
            "IPv4Address": "192.168.20.4/24",
            "IPv6Address": ""
        },
        "46732b3736b6749aeb6e07c425d0bc1a3e19962f4aaff4c282d80e982425256b": {
            "Name": "web4",
            "EndpointID": "3299d22ad2b167390fccd50697c7e800dc06ae41d0b1b09f2bd4
6192315d1dcf",
            "MacAddress": "02:42:c0:a8:14:05",
            "IPv4Address": "192.168.20.5/24",
            "IPv6Address": ""
        },
        "4b39d634d08e0ca83b8e9cc10f48a765b3cde6bf1162bf463a57f92d597d88e4": {
            "Name": "web6",
            "EndpointID": "4ca2de3c37ce6b88823a5beec0b667226c2d0b793eaaa4cc75b6
55ee50806f70",
            "MacAddress": "02:42:c0:a8:14:07",
            "IPv4Address": "192.168.20.7/24",
            "IPv6Address": ""
        },
        "5f97e75d063890e90e542082da87143d167a29fb2e0048cc12968127ac3fed3": {
            "Name": "web8",
            "EndpointID": "a1f9f17236dbe13f66d9cf08cce8d5499cba430fa01791d9f0b6
fbcf37144842",
            "MacAddress": "02:42:c0:a8:14:09",
            "IPv4Address": "192.168.20.9/24",
            "IPv6Address": ""
        },
    ],
    "NetworkSettings": {
        "Bridge": "bridge",
        "ContainerID": "4ca2de3c37ce6b88823a5beec0b667226c2d0b793eaaa4cc75b6
55ee50806f70",
        "GlobalIPv4Address": "192.168.20.7",
        "GlobalIPv6Address": null,
        "Gateway": "192.168.20.1",
        "IPAddress": "192.168.20.7",
        "IPPrefixLen": 24,
        "Links": null,
        "MacAddress": "02:42:c0:a8:14:07",
        "NetworkMode": "bridge",
        "Ports": {
            "80/tcp": [
                {
                    "HostIp": "0.0.0.0",
                    "HostPort": 80
                }
            ]
        },
        "SecondaryIPAddresses": null,
        "SecondaryIPv6Addresses": null,
        "Subnet": "192.168.20.0/24"
    }
}
```

```

        "6bf9042bbcf4a39f30c51a532af5192b3439fd6c3f00665ccb513f973f2e5d5e": {
            "Name": "web5",
            "EndpointID": "19d529e920b625a5cd629a5f70483b9fbc1f6b3eacfda424425
1fbcd7154092",
            "MacAddress": "02:42:c0:a8:14:06",
            "IPv4Address": "192.168.20.6/24",
            "IPv6Address": ""
        },
        "88ab1e4baf249353e82ea1c174855071c999f21cd6736cbe65eabd1fe62c6b87": {
            "Name": "web2",
            "EndpointID": "dc213d8adf45797095cc4a27ed97650fd44179bea8a5b052e563
4866a6b95427",
            "MacAddress": "02:42:c0:a8:14:03",
            "IPv4Address": "192.168.20.3/24",
            "IPv6Address": ""
        },
        "baa09692f53864bbb556127337c53ea0c93152e6ddac875f154a8a95ff65e538": {
            "Name": "web1",
            "EndpointID": "45de6cbfaaa03e47c0e00152679879a24402a0980a06d5dcf150
828d62390fa4",
            "MacAddress": "02:42:c0:a8:14:02",
            "IPv4Address": "192.168.20.2/24",
            "IPv6Address": ""
        }
    },
    "Options": {},
    "Labels": {
        "com.docker.compose.network": "red_servicios",
        "com.docker.compose.project": "p1",
        "com.docker.compose.version": "2.23.3"
    }
}
]

```

- Comprobar conectividad entre los distintos contenedores.

Conectividad del contenedor web1 al web2:

```

2025-03-20 13:03:37 ✎ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
o ➔ docker compose exec web1 /bin/bash
root@d52718ecab6a:/var/www/html# ping 192.168.10.3
PING 192.168.10.3 (192.168.10.3) 56(84) bytes of data.
64 bytes from 192.168.10.3: icmp_seq=1 ttl=64 time=0.091 ms
64 bytes from 192.168.10.3: icmp_seq=2 ttl=64 time=0.056 ms
64 bytes from 192.168.10.3: icmp_seq=3 ttl=64 time=0.078 ms
64 bytes from 192.168.10.3: icmp_seq=4 ttl=64 time=0.081 ms
64 bytes from 192.168.10.3: icmp_seq=5 ttl=64 time=0.078 ms
64 bytes from 192.168.10.3: icmp_seq=6 ttl=64 time=0.079 ms
64 bytes from 192.168.10.3: icmp_seq=7 ttl=64 time=0.076 ms
64 bytes from 192.168.10.3: icmp_seq=8 ttl=64 time=0.062 ms
64 bytes from 192.168.10.3: icmp_seq=9 ttl=64 time=0.038 ms
64 bytes from 192.168.10.3: icmp_seq=10 ttl=64 time=0.038 ms
64 bytes from 192.168.10.3: icmp_seq=11 ttl=64 time=0.060 ms
64 bytes from 192.168.10.3: icmp_seq=12 ttl=64 time=0.056 ms
64 bytes from 192.168.10.3: icmp_seq=13 ttl=64 time=0.078 ms
64 bytes from 192.168.10.3: icmp_seq=14 ttl=64 time=0.077 ms
64 bytes from 192.168.10.3: icmp_seq=15 ttl=64 time=0.061 ms
64 bytes from 192.168.10.3: icmp_seq=16 ttl=64 time=0.063 ms
64 bytes from 192.168.10.3: icmp_seq=17 ttl=64 time=0.080 ms
64 bytes from 192.168.10.3: icmp_seq=18 ttl=64 time=0.078 ms
64 bytes from 192.168.10.3: icmp_seq=19 ttl=64 time=0.061 ms
64 bytes from 192.168.10.3: icmp_seq=20 ttl=64 time=0.061 ms
64 bytes from 192.168.10.3: icmp_seq=21 ttl=64 time=0.078 ms
64 bytes from 192.168.10.3: icmp_seq=22 ttl=64 time=0.047 ms
64 bytes from 192.168.10.3: icmp_seq=23 ttl=64 time=0.059 ms
64 bytes from 192.168.10.3: icmp_seq=24 ttl=64 time=0.079 ms
^C
--- 192.168.10.3 ping statistics ---
24 packets transmitted, 24 received, 0% packet loss, time 23547ms
rtt min/avg/max/mdev = 0.038/0.067/0.091/0.013 ms

```

Conecvidad del contenedor web1 al web3:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.4
PING 192.168.10.4 (192.168.10.4) 56(84) bytes of data.
64 bytes from 192.168.10.4: icmp_seq=1 ttl=64 time=0.094 ms
64 bytes from 192.168.10.4: icmp_seq=2 ttl=64 time=0.056 ms
64 bytes from 192.168.10.4: icmp_seq=3 ttl=64 time=0.059 ms
64 bytes from 192.168.10.4: icmp_seq=4 ttl=64 time=0.060 ms
64 bytes from 192.168.10.4: icmp_seq=5 ttl=64 time=0.084 ms
64 bytes from 192.168.10.4: icmp_seq=6 ttl=64 time=0.078 ms
64 bytes from 192.168.10.4: icmp_seq=7 ttl=64 time=0.051 ms
64 bytes from 192.168.10.4: icmp_seq=8 ttl=64 time=0.057 ms
64 bytes from 192.168.10.4: icmp_seq=9 ttl=64 time=0.094 ms
64 bytes from 192.168.10.4: icmp_seq=10 ttl=64 time=0.061 ms
64 bytes from 192.168.10.4: icmp_seq=11 ttl=64 time=0.057 ms
64 bytes from 192.168.10.4: icmp_seq=12 ttl=64 time=0.077 ms
^C
--- 192.168.10.4 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11301ms
rtt min/avg/max/mdev = 0.051/0.069/0.094/0.014 ms
```

Conecvidad del contenedor web1 al web4:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.5
PING 192.168.10.5 (192.168.10.5) 56(84) bytes of data.
64 bytes from 192.168.10.5: icmp_seq=1 ttl=64 time=0.091 ms
64 bytes from 192.168.10.5: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 192.168.10.5: icmp_seq=3 ttl=64 time=0.051 ms
64 bytes from 192.168.10.5: icmp_seq=4 ttl=64 time=0.052 ms
64 bytes from 192.168.10.5: icmp_seq=5 ttl=64 time=0.045 ms
64 bytes from 192.168.10.5: icmp_seq=6 ttl=64 time=0.061 ms
64 bytes from 192.168.10.5: icmp_seq=7 ttl=64 time=0.060 ms
64 bytes from 192.168.10.5: icmp_seq=8 ttl=64 time=0.060 ms
64 bytes from 192.168.10.5: icmp_seq=9 ttl=64 time=0.044 ms
^C
--- 192.168.10.5 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8203ms
rtt min/avg/max/mdev = 0.044/0.058/0.091/0.013 ms
```

Conecividad del contenedor web1 al web5:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.6
PING 192.168.10.6 (192.168.10.6) 56(84) bytes of data.
64 bytes from 192.168.10.6: icmp_seq=1 ttl=64 time=0.090 ms
64 bytes from 192.168.10.6: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 192.168.10.6: icmp_seq=3 ttl=64 time=0.082 ms
64 bytes from 192.168.10.6: icmp_seq=4 ttl=64 time=0.105 ms
64 bytes from 192.168.10.6: icmp_seq=5 ttl=64 time=0.082 ms
64 bytes from 192.168.10.6: icmp_seq=6 ttl=64 time=0.084 ms
^C
--- 192.168.10.6 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5110ms
rtt min/avg/max/mdev = 0.048/0.081/0.105/0.017 ms
```

Conecividad del contenedor web1 al web6:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.7
PING 192.168.10.7 (192.168.10.7) 56(84) bytes of data.
64 bytes from 192.168.10.7: icmp_seq=1 ttl=64 time=0.091 ms
64 bytes from 192.168.10.7: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 192.168.10.7: icmp_seq=3 ttl=64 time=0.080 ms
64 bytes from 192.168.10.7: icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from 192.168.10.7: icmp_seq=5 ttl=64 time=0.080 ms
64 bytes from 192.168.10.7: icmp_seq=6 ttl=64 time=0.078 ms
^C
--- 192.168.10.7 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5108ms
rtt min/avg/max/mdev = 0.048/0.076/0.091/0.013 ms
```

Conecividad del contenedor web1 al web7:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.8
PING 192.168.10.8 (192.168.10.8) 56(84) bytes of data.
64 bytes from 192.168.10.8: icmp_seq=1 ttl=64 time=0.093 ms
64 bytes from 192.168.10.8: icmp_seq=2 ttl=64 time=0.057 ms
64 bytes from 192.168.10.8: icmp_seq=3 ttl=64 time=0.054 ms
64 bytes from 192.168.10.8: icmp_seq=4 ttl=64 time=0.061 ms
64 bytes from 192.168.10.8: icmp_seq=5 ttl=64 time=0.083 ms
^C
--- 192.168.10.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4099ms
rtt min/avg/max/mdev = 0.054/0.069/0.093/0.015 ms
```

Conecvidad del contenedor web1 al web8:

```
root@d52718ecab6a:/var/www/html# ping 192.168.10.9
PING 192.168.10.9 (192.168.10.9) 56(84) bytes of data.
64 bytes from 192.168.10.9: icmp_seq=1 ttl=64 time=0.092 ms
64 bytes from 192.168.10.9: icmp_seq=2 ttl=64 time=0.207 ms
64 bytes from 192.168.10.9: icmp_seq=3 ttl=64 time=0.062 ms
64 bytes from 192.168.10.9: icmp_seq=4 ttl=64 time=0.083 ms
64 bytes from 192.168.10.9: icmp_seq=5 ttl=64 time=0.079 ms
^C
--- 192.168.10.9 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4104ms
rtt min/avg/max/mdev = 0.062/0.104/0.207/0.052 ms
```

Conecvidad del contenedor web4 al web8:

```
2025-03-20 13:16:22 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1
○ o ➔ docker compose exec web4 /bin/bash
root@b3f193d9a159:/var/www/html# ping 192.168.10.9
PING 192.168.10.9 (192.168.10.9) 56(84) bytes of data.
64 bytes from 192.168.10.9: icmp_seq=1 ttl=64 time=0.088 ms
64 bytes from 192.168.10.9: icmp_seq=2 ttl=64 time=0.080 ms
64 bytes from 192.168.10.9: icmp_seq=3 ttl=64 time=0.077 ms
64 bytes from 192.168.10.9: icmp_seq=4 ttl=64 time=0.061 ms
^C
--- 192.168.10.9 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3092ms
rtt min/avg/max/mdev = 0.061/0.076/0.088/0.009 ms
```

2.5. B5. Pruebas Básicas

- Acceder a la página web de cada contenedor usando su dirección IP y verificar que muestra la información correcta



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.2



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.3



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.4



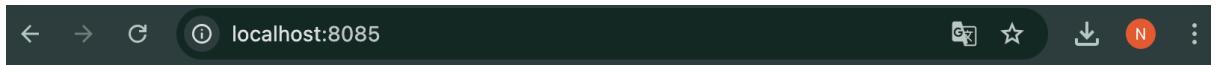
Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.5



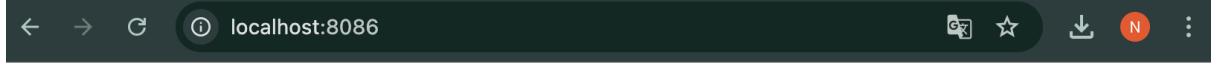
Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.6



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.7



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.8



Práctica 1 SWAP - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.20.9

2.6. Análisis propuesta IA

Con este apartado voy a tratar de **analizar los resultados propuestos por la IA** para este primer apartado de tareas básicas. He preguntado a la IA escribiendo directamente el enunciado de los diferentes apartados para que me mostrara paso a paso lo que debe de incluir cada uno, [pulsa aquí para acceder al chat](#)

De forma resumida, la IA me ha proporcionado soluciones muy similares a las mías aunque con algunas diferencias como:

1. Contenido index.php: Prefiero mi solución ya que, hago uso de html y php de forma que se aprecie una separación clara entre cada una de las partes. Además, como estoy cursando Tecnologías Web este mismo año estamos trabajando con lo mismo, lo que me parece mucho mejor que la solución que me ha proporcionado la IA.

2. Imagen base de linux: La IA me ha sugerido usar Ubuntu, pero en mi caso me encuentro más cómoda trabajando con Debian, por lo que, prefiero mi opción.
3. Instalación Apache, PHP y herramientas de red: La solución que me ha propuesto la IA no es del todo completa ya que, falta instalar herramientas de red fundamentales (net-tools) e incluso módulos php (libapache2-mod-php).
4. Copiar el contenido del directorio web_mmnuria al directorio raíz de Apache: Lo que me está sugiriendo la IA no es necesario ya que, como estamos usando volúmenes para montar el directorio, con la línea añadida en cada contenedor

```
volumes:
| - ./web_mmnuria:/var/www/html
```

es más que suficiente para montar el directorio web_mmnuria (local) en el directorio /var/www/html de cada contenedor.

5. Archivo docker-compose.yml: La solución propuesta por la IA está incompleta por un lado faltan las direcciones ip asignadas a cada red para cada contenedor y por otro lado, los puertos de cada uno de los contenedores.

```
networks:
| red_web:
| | ipv4_address: 192.168.10.2
| red_servicios:
| | ipv4_address: 192.168.20.2
ports:
| - "8080:80"
```

3. Desarrollo tareas avanzadas

Para resolver este apartado he optado por la creación de una carpeta nueva “P1-ADVANCED” en la que se montan de nuevo todos los contenedores con la resolución de cada una de las tareas paso a paso. Por cada servidor web monto 2 contenedores de forma que se pueda observar su correcto funcionamiento con una doble comprobación.

3.1. A1: Personalización del Dockerfile

- Modificar el Dockerfile para incluir configuraciones personalizadas de Apache o PHP

El nuevo Dockerfile mostrado a continuación para el servidor Apache incluye solamente las partes en las que se han añadido configuraciones personalizadas. Para ver el archivo completo revisarlo en “P1-ADVANCED/dockerfiles/DockerfileApache_mmnuria_advanced”

```

# Configuración personalizada de PHP
RUN echo "max_execution_time = 120" >> /etc/php/7.4/apache2/php.ini && \
    echo "upload_max_filesize = 20M" >> /etc/php/7.4/apache2/php.ini && \
    echo "post_max_size = 21M" >> /etc/php/7.4/apache2/php.ini && \
    echo "memory_limit = 256M" >> /etc/php/7.4/apache2/php.ini
PHP
APACHE

# Configuración personalizada de Apache
COPY ./config/apache/custom-apache.conf /etc/apache2/conf-available/
RUN a2enconf custom-apache

# Configurar el directorio raíz de Apache para priorizar index.php
RUN sed -i 's/DirectoryIndex index.html DirectoryIndex index.php index.html/' /etc/apache2/mods-enabled/dir.conf

# Configurar virtual hosts
COPY ./config/apache/vhosts.conf /etc/apache2/sites-available/
RUN a2ensite vhosts

# Establecer zona horaria
RUN ln -sf /usr/share/zoneinfo/Europe/Madrid /etc/localtime
OTRAS

# Directorio de trabajo
WORKDIR /var/www/html

# Configurar permisos correctos
RUN chown -R www-data:www-data /var/www/html /var/log/apache2

# Configurar rotación de logs
COPY ./config/apache/apache-logrotate /etc/logrotate.d/apache2

# Copiar scripts de mantenimiento
COPY ./scripts/mantenimiento.sh /usr/local/bin/
RUN chmod +x /usr/local/bin/mantenimiento.sh

```

Las configuraciones personalizadas son:

1. PHP: se pueden ver resaltadas en color rosa:

- `max_execution_time = 120`: Aumenta el tiempo máximo de ejecución de scripts PHP a 120 segundos
- `upload_max_filesize = 20M`: Permite subir archivos de hasta 20MB
- `post_max_size = 21M`: Establece el tamaño máximo de datos POST a 21MB (ligeramente mayor que `upload_max_filesize`)
- `memory_limit = 256M`: Asigna hasta 256MB de memoria para scripts PHP

2. APACHE: se pueden ver resaltadas de color verde:

- Inclusión de un archivo de configuración personalizado **custom-apache.conf**

```

2025-03-23 19:56:43 ✎ Nurias-MacBook-Pro
[○ → cat custom-apache.conf
<IfModule mpm_prefork_module>
    StartServers          5
    MinSpareServers       5
    MaxSpareServers      10
    MaxRequestWorkers    150
    MaxConnectionsPerChild 0
</IfModule>

ServerTokens Prod
ServerSignature Off
TraceEnable Off

```

Como se aprecia en la imagen, éste archivo configura varios aspectos del rendimiento y la seguridad de Apache:

- **Configuración de MPM Prefork:**

- `StartServers 5`: Inicia Apache con 5 procesos hijo
- `MinSpareServers 5`: Mantiene al menos 5 procesos hijo en espera
- `MaxSpareServers 10`: No mantiene más de 10 procesos hijo en espera
- `MaxRequestWorkers 150`: Permite hasta 150 conexiones simultáneas
- `MaxConnectionsPerChild 0`: Cada proceso hijo puede manejar un número ilimitado de conexiones antes de reciclarse

- **Configuración de seguridad:**

- `ServerTokens Prod`: Limita la información expuesta en cabeceras HTTP mostrando solo "Apache" sin versión
- `ServerSignature Off`: Desactiva la firma de Apache en páginas de error generadas
- `TraceEnable Off`: Desactiva el método HTTP TRACE para prevenir ataques XST (Cross-Site Tracing)

- Priorización de **index.php** sobre **index.html** en el `DirectoryIndex`
- Configuración de hosts virtuales a través de **vhosts.conf**

```
2025-03-23 19:56:50 Nurias-MacBook-Pro in ~/Desks
[○ → cat vhosts.conf
<VirtualHost *:80>
    ServerName local.web
    DocumentRoot /var/www/html

    <Directory /var/www/html>
        Options -Indexes +FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

En este archivo podemos ver que se configura un host virtual en Apache donde se distribuye:

- **Configuración básica del VirtualHost:**

- <VirtualHost *:80>: Crea un host virtual que responde en el puerto 80 para todas las IPs
- ServerName local.web: Define el nombre del servidor como "local.web"
- DocumentRoot /var/www/html: Establece el directorio raíz de documentos

- **Configuración del directorio principal:**

- <Directory /var/www/html>: Aplica reglas al directorio raíz
- Options -Indexes +FollowSymLinks: Desactiva el listado de directorios pero permite seguir enlaces simbólicos
- AllowOverride All: Permite que los archivos .htaccess modifiquen la configuración
- Require all granted: Concede acceso a todos los usuarios

- **Configuración de logs:**

- ErrorLog
\${APACHE_LOG_DIR}/error.log: Define la ubicación del registro de errores
 - CustomLog
\${APACHE_LOG_DIR}/access.log
combined: Define la ubicación y formato del registro de acceso
- Configuración de rotación de logs personalizada

3. Otras mejoras operativas:

- Configuración de zona horaria a Europa/Madrid
- Inclusión de un script de mantenimiento (profundizaré en él más adelante)

3.2. A2. Creación de contenedores con otros servidores web

- Crear contenedores con otros servidores web (nginx, lighttpd, etc.)

En primer lugar, veremos la resolución de la tarea para el contenedor con el **servidor web Lighttpd**:

1. Dockerfile:

```

2025-03-24 13:26:06 ⓘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADVAN
CED/dockerfiles
[○ → cat DockerfileLighttpd_mmnuria
FROM debian:bullseye

# Instalar Lighttpd y PHP con FPM
RUN apt-get update && apt-get install -y \
    lighttpd \
    php7.4-fpm \
    php7.4-cgi \
    php7.4-mysql \
    php7.4-gd \
    php7.4-curl \
    php7.4-xml \
    php7.4-mbstring \
    iputils-ping \
    iproute2 \
    net-tools \
    curl \
    htop \
    logrotate \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

# Configurar Lighttpd para usar PHP
RUN lighttpd-enable-mod fastcgi && \
    lighttpd-enable-mod fastcgi-php

# Copiar configuración personalizada
COPY ./config/lighttpd/lighttpd.conf /etc/lighttpd/lighttpd.conf

# Modificar la configuración de PHP-FPM para usar el socket en /tmp
RUN sed -i 's|listen = /run/php/php7.4-fpm.sock|listen = /tmp/php-fastcgi.sock|g' /etc/php/7.4/f
pm/pool.d/www.conf && \
    mkdir -p /var/log/lighttpd && \
    chown www-data:www-data /var/log/lighttpd

# Crear script de inicio
COPY ./scripts/inicio-lighttpd.sh /start.sh
RUN chmod +x /start.sh

# Directorio de trabajo
WORKDIR /var/www/html

# Configurar permisos
RUN chown -R www-data:www-data /var/www/html

# Exponer puerto
EXPOSE 80

# Iniciar servicios
CMD ["/start.sh"]

```

Configura un contenedor con Lighttpd y PHP-FPM en lugar de Apache.

● Instalación de paquetes

- Instala Lighttpd como servidor web (más ligero que Apache)
- Instala PHP 7.4 con FPM (FastCGI Process Manager) y varios módulos necesarios
- Incluye herramientas de red y monitoreo similares a tu Dockerfile de Apache

● Configuración de PHP-FPM

La **parte más importante** es esta modificación:

```
# Modificar la configuración de PHP-FPM para usar el socket en /tmp
RUN sed -i 's|listen = /run/php/php7.4-fpm.sock|listen = /tmp/php-fastcgi.sock|g' /etc/php/7.4/fpm/pool.d/www.conf && \
```

He tenido que añadir esta configuración porque:

1. Conexión entre Lighttpd y PHP: Lighttpd y PHP-FPM se **comunican a través de un socket**. Por defecto, PHP-FPM usa `'/run/php/php7.4-fpm.sock'`, pero en este caso se cambia a `'/tmp/php-fastcgi.sock'`.
2. Accesibilidad del socket: El socket necesita estar en una ubicación donde Lighttpd pueda acceder. La carpeta `/tmp` es más accesible y tiene permisos adecuados dentro del contenedor.
3. FastCGI vs. mod_php: A diferencia de Apache, que puede usar mod_php (integrado directamente), **Lighttpd utiliza FastCGI para comunicarse con PHP, lo que requiere esta configuración específica del socket**.

- **Otras partes importantes:**

- Habilito los módulos fastcgi y fastcgi-php en Lighttpd
- Copio una configuración personalizada de Lighttpd
- Uso un **script de inicio personalizado**:

Este script crea el directorio para logs y asigna permisos, inicia el servicio PHP-FPM (7.4), verifica que se haya creado el socket, ajusta los permisos (del socket) para permitir la comunicación e inicia el servidor web Lighttpd en primer plano.

Su **función principal** es asegurar que tanto el servidor web como PHP estén correctamente configurados y puedan comunicarse entre sí antes de iniciar el servicio.

```

2025-03-23 20:31:49 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO
/SEGUNDO CUATRI/SWAP/P1-ADVANCED/scripts
|o ➔ cat inicio-lighttpd.sh
#!/bin/bash

# Asegurar que los directorios de logs existan
mkdir -p /var/log/lighttpd
chown www-data:www-data /var/log/lighttpd

# Iniciar PHP-FPM
service php7.4-fpm start

# Verificar que PHP-FPM se haya iniciado correctamente
echo "Verificando PHP-FPM..."
for i in {1..5}; do
    if [ -S "/tmp/php-fastcgi.sock" ]; then
        echo "Socket de PHP-FPM encontrado!"
        break
    fi
    echo "Esperando a que PHP-FPM cree el socket ($i/5)..."
    sleep 1
done

# Verificar que el socket existe
if [ ! -S "/tmp/php-fastcgi.sock" ]; then
    echo "ERROR: PHP-FPM no creó el socket en /tmp/php-fastcgi.sock"
    echo "Contenido de /tmp:"
    ls -la /tmp
    echo "Estado de PHP-FPM:"
    service php7.4-fpm status
    exit 1
fi

# Comprobar los permisos del socket
ls -la /tmp/php-fastcgi.sock
chmod 666 /tmp/php-fastcgi.sock

# Iniciar Lighttpd en primer plano
echo "Iniciando Lighttpd..."
lighttpd -D -f /etc/lighttpd/lighttpd.conf

```

- Configuro permisos y directorios de manera similar al contenedor Apache

2. Archivo docker-compose:

Defino dos servicios web Lighttpd de forma muy similar a lo realizado en el primer apartado de las tareas básicas, pero cambiando el puerto del host y direcciones ips, para evitar conflictos.

```

lighttpd1:
  build:
    context: .
    dockerfile: ./dockerfiles/DockerfileLighttpd_mmnuria
  image: mmnuria-lighttpd:latest
  volumes:
    - ./web_mmnuria:/var/www/html
    - ./config/lighttpd/lighttpd.conf:/etc/lighttpd/lighttpd.conf
  networks:
    red_web:
      ipv4_address: 192.168.100.30
    red_servicios:
      ipv4_address: 192.168.200.30
  ports:
    - "8092:80"
  restart: unless-stopped

lighttpd2:
  image: mmnuria-lighttpd:latest
  volumes:
    - ./web_mmnuria:/var/www/html
    - ./config/lighttpd/lighttpd.conf:/etc/lighttpd/lighttpd.conf
  networks:
    red_web:
      ipv4_address: 192.168.100.31
    red_servicios:
      ipv4_address: 192.168.200.31
  ports:
    - "8093:80"
  restart: unless-stopped

```

3. Archivo lighttpd.conf:

Establezco un servidor web básico con soporte PHP a través de FastCGI donde se habilitan diferentes módulos, se configura directorio raíz, puerto, interfaces, etc. Además, se establecen un orden en la búsqueda de los archivos de índice siendo el de mayor prioridad el index.php (lo tenemos en el directorio web_mmnuria copiado en el directorio raíz). Y por último, configuraciones de PHP y logs para errores y accesos.

```

2025-03-23 20:27:20 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO
/SEGUNDO CUATRI/SWAP/P1-ADVANCED/config/lighttpd
[O → cat lighttpd.conf
server.modules =
  "mod_access",
  "mod_alias",
  "mod_compress",
  "mod_redirect",
  "mod_fastcgi"
)

server.document-root = "/var/www/html"
server.port = 80
server.bind = "0.0.0.0"
server.dir-listing = "enable"

index-file.names = ( "index.php", "index.html", "index.htm" )

# Configuración de PHP con FastCGI
fastcgi.server = ( ".php" =>
  ( "localhost" =>
    (
      "socket" => "/tmp/php-fastcgi.sock",
      "broken-scriptfilename" => "enable"
    )
  )
)

# Logs
server.errorlog = "/var/log/lighttpd/error.log"
# Nota: accesslog.filename da un warning, usamos la forma correcta:
accesslog.filename = "/var/log/lighttpd/access.log"

```

Finalmente, veremos la resolución de la tarea para el contenedor con el **servidor web Nginx**:

1. Dockerfile:

```

2025-03-24 13:26:12 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADVAN
CED/dockerfiles
[○ → cat DockerfileNginx_mmnuria
FROM debian:bullseye

# Instalar Nginx y PHP-FPM
RUN apt-get update && apt-get install -y \
    nginx \
    php-fpm \
    php-mysql \
    php-gd \
    php-curl \
    php-xml \
    php-mbstring \
    iputils-ping \
    iproute2 \
    net-tools \
    curl \
    htop \
    logrotate \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

# Copiar el archivo nginx.conf directamente al contenedor
COPY ./config/nginx/nginx.conf /etc/nginx/nginx.conf

# Configurar PHP
RUN echo "cgi.fix_pathinfo=0" >> /etc/php/7.4/fpm/php.ini && \
    echo "upload_max_filesize = 20M" >> /etc/php/7.4/fpm/php.ini && \
    echo "post_max_size = 21M" >> /etc/php/7.4/fpm/php.ini && \
    echo "memory_limit = 256M" >> /etc/php/7.4/fpm/php.ini

# Configurar PHP-FPM para usar TCP en lugar de socket
RUN sed -i 's|listen = /run/php/php7.4-fpm.sock|listen = 127.0.0.1:9000|g' /etc/php/7.4/fpm/pool
.d/www.conf

WORKDIR /var/www/html

# Configurar permisos para que Nginx y PHP puedan acceder
RUN chown -R www-data:www-data /var/www/html && chmod -R 755 /var/www/html

# Copiar script de inicio
COPY ./scripts/inicio-nginx.sh /usr/local/bin/start.sh
RUN chmod +x /usr/local/bin/start.sh

# Exponer puertos
EXPOSE 80 443

# Iniciar Nginx y PHP-FPM
CMD ["/usr/local/bin/start.sh"]

```

Como se puede observar, este archivo dockerfile es similar al anterior (Lighttpd), pero como en este caso se configura el servidor web Nginx existen algunas diferencias claras y necesarias con respecto a Lighttpd, estas son:

- Servidor web diferente (obviamente).
- Paquete PHP necesario, en este caso con el php-fpm (paquete general) es suficiente.
- Configuración del PHP-FFPM para usar TCP en vez de socket que es lo que se necesitaba para Lighttpd (he estado realizando diferentes pruebas y la única manera para que los servidores funcionen es cambiando uno a TCP y otro con socket).
- Script de inicio:

```

2025-03-23 20:56:41 ⚡ Nurias-MacBook-Pro
TRI/SWAP/P1-ADVANCED/scripts
[○ → cat inicio-nginx.sh
#!/bin/bash
# Iniciar PHP-FPM
service php7.4-fpm start

# Iniciar Nginx en primer plano
nginx -g "daemon off;"
```

El script simplemente:

1. Inicia PHP-FPM en segundo plano
2. Ejecuta Nginx en primer plano con `daemon off` para mantener el contenedor activo

Es **necesario** porque Docker requiere un proceso en primer plano y necesitamos ejecutar dos servicios (Nginx y PHP-FPM) en el mismo contenedor.

- A diferencia con Lighttpd no necesitamos habilitar ningún módulo ni configuración php adicional, ni logs.

2. Archivo docker-compose:

```

nginx1:
  build:
    context: .
    dockerfile: ./dockerfiles/DockerfileNginx_mmnuria
  image: mmnuria-nginx:latest
  volumes:
    - ./web_mmnuria:/var/www/html
    - ./config/nginx/nginx.conf:/etc/nginx/nginx.conf
  networks:
    red_web:
      ipv4_address: 192.168.100.20
    red_servicios:
      ipv4_address: 192.168.200.20
  ports:
    - "8090:80"
  restart: unless-stopped

nginx2:
  image: mmnuria-nginx:latest
  volumes:
    - ./web_mmnuria:/var/www/html
    - ./config/nginx/nginx.conf:/etc/nginx/nginx.conf
  networks:
    red_web:
      ipv4_address: 192.168.100.21
    red_servicios:
      ipv4_address: 192.168.200.21
  ports:
    - "8091:80"
  restart: unless-stopped
```

En la captura se puede apreciar la configuración de dos servicios web Nginx de forma muy similar a lo realizado con Lighttpd, pero cambiando el puerto del host y direcciones ips, para evitar conflictos.

3. Archivo nginx.conf:

```
2025-03-23 23:01:52 🏠 Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUAT
RI/SWAP/P1-ADVANCED/config/nginx
○ ➔ cat nginx.conf
worker_processes 1;
pid /var/run/nginx.pid;

events {}

http {
    include      /etc/nginx/mime.types;
    default_type application/octet-stream;

    server {
        listen 80;
        server_name localhost;

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

        location / {
            try_files $uri $uri/ =404;
        }

        location ~ \.php$ {
            include fastcgi_params;
            fastcgi_pass 127.0.0.1:9000;
            fastcgi_index index.php;
            fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        }

        error_log /var/log/nginx/error.log warn;
        access_log /var/log/nginx/access.log;
    }
}
```

Este archivo define la configuración básica del host virtual en Nginx:

1. Escucha en puerto 80
2. Establece documentos raíz en /var/www/html
3. Define orden de archivos de índice (primero .php, luego .html)
4. Configuración para archivos estáticos con try_files
5. Procesamiento de PHP mediante fastcgi conectando a 127.0.0.1:9000 (TCP)
6. Ubicación de archivos de logs de errores y acceso

La **Línea clave** es `fastcgi_pass 127.0.0.1:9000` que conecta Nginx con PHP-FPM usando TCP en lugar de un socket.

3.3. A3. Gestión Avanzada de Redes

- Configurar reglas específicas de enrutamiento o restricciones de acceso entre las dos redes red_web y red_servicios.

Para establecer reglas específicas de enrutamiento o restricciones de acceso entre las dos redes he creado un contendedor llamado “network_manager” para poder administrar la red y a través de un script llamado “configuracion_redes.sh” lo ejecuto en el contenedor para configurar las reglas de forma que nos queda:

1. docker-compose: Contenedor de gestión de redes

```
# Contenedor para gestión de redes
network_manager:
  image: alpine:latest
  container_name: network_manager
  cap_add:
    - NET_ADMIN
  privileged: true
  volumes:
    - ./scripts:/scripts
  entrypoint: ["/bin/sh", "/scripts/configuracion-redes.sh"]
  networks:
    red_web:
      ipv4_address: 192.168.100.2
    red_servicios:
      ipv4_address: 192.168.200.2
  restart: unless-stopped
```

Lo único destacable de lo que ya hemos hecho en los apartados anteriores es el script donde se configuran las redes.

2. Script para la configuración de las reglas:

```

2025-03-25 09:27:03 ✎ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADVA
CED/scripts
[o ➔ cat configuracion-redes.sh
#!/bin/bash
set -e

echo "Configurando reglas de red..."

# Instalar las herramientas necesarias (esto es para el host, no para el contenedor)
apk add --no-cache iptables iproute2 iputils-ping net-tools

# Habilitar el reenvío de IP en el host (para permitir el enrutamiento entre las redes)
echo "Habilitando reenvío de IP..."
sysctl -w net.ipv4.ip_forward=1

# Configurar reglas para red_web
echo "Configurando reglas para red_web (192.168.100.0/24)...""

# Permitir que los servidores nginx solo reciban tráfico en puerto 80
iptables -A FORWARD -d 192.168.100.20/32 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -d 192.168.100.20/32 -j DROP
iptables -A FORWARD -d 192.168.100.21/32 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -d 192.168.100.21/32 -j DROP

# Configurar reglas para red_servicios
echo "Configurando reglas para red_servicios (192.168.200.0/24)...""

# Restringir tráfico entre servidores apache y lighttpd
iptables -A FORWARD -s 192.168.200.10/32 -d 192.168.200.30/32 -j DROP
iptables -A FORWARD -s 192.168.200.10/32 -d 192.168.200.31/32 -j DROP
iptables -A FORWARD -s 192.168.200.11/32 -d 192.168.200.30/32 -j DROP
iptables -A FORWARD -s 192.168.200.11/32 -d 192.168.200.31/32 -j DROP

# Asegurarse de que el tráfico entre las redes esté permitido (si es necesario)
# Por ejemplo, permitir tráfico entre las redes red_web y red_servicios en puertos específicos
iptables -A FORWARD -i br_red_web -o br_red_svs -j ACCEPT
iptables -A FORWARD -i br_red_svs -o br_red_web -j ACCEPT

echo "Configuración de red completada."

# Mantener el contenedor ejecutándose
tail -f /dev/null

```

- 1. Instalación de herramientas de red:** Instala las herramientas necesarias (iptables, iproute2, iputils-ping, net-tools) en el host para gestionar las redes y las reglas de tráfico.
- 2. Habilita el reenvío de IP:** Activa el reenvío de IP en el host para permitir el enrutamiento entre las redes Docker `red_web` y `red_servicios`.
- 3. Configura reglas de iptables:**
 - Para `red_web`: Permite tráfico solo en el puerto 80 entre los servidores nginx.
 - Para `red_servicios`: Restringe el tráfico entre los servidores apache y lighttpd para evitar que se comuniquen entre sí.
- 4. Mantener el contenedor en ejecución:** Al final, el script tiene `tail -f /dev/null` para mantener el contenedor activo después de aplicar las configuraciones.

3.4. A4. Automatización con Scripts

- Crear scripts para tareas de mantenimiento automatizado, como limpieza de logs, monitoreo de la salud del contenedor, o actualizaciones automáticas de paquetes.

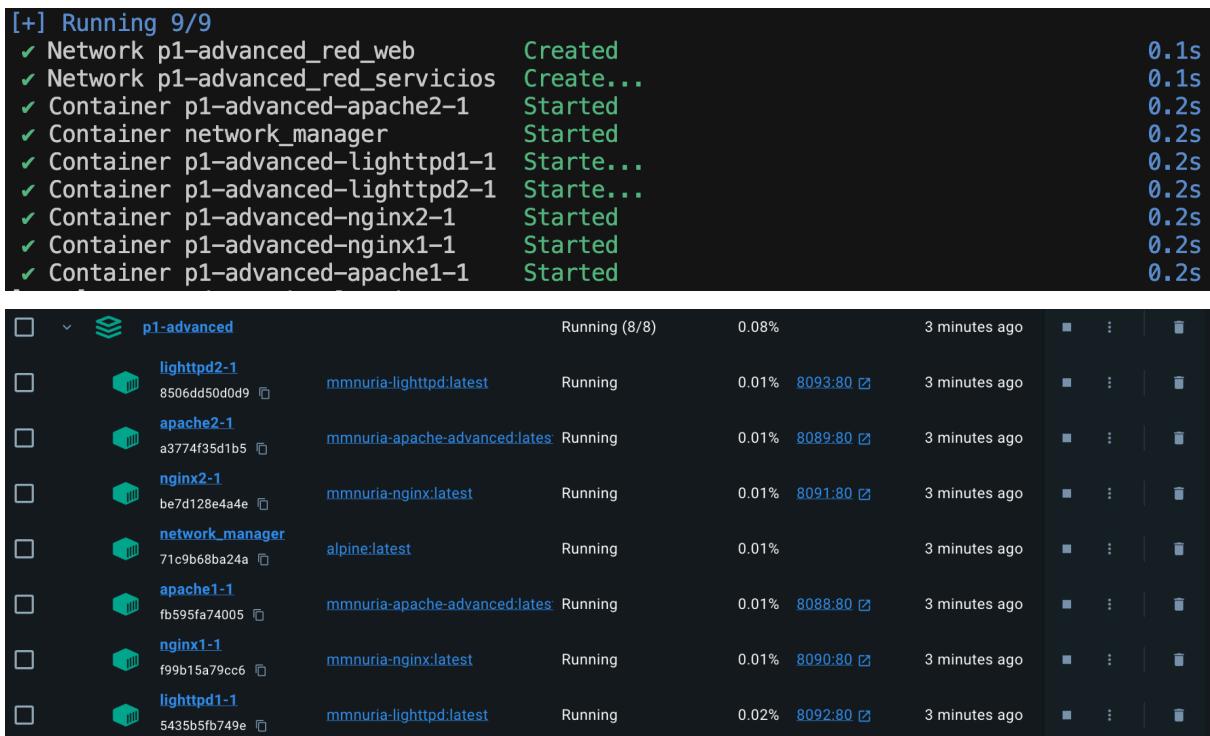
Para resolver este apartado he creado el archivo **mantenimiento.sh** en el que se realizan diferentes tareas como limpieza de logs o estados de los servicios. Al ser un archivo con bastantes líneas de código mostraré solamente el resultado positivo de su ejecución y el contenido del informe generado del estado de los servicios.

```
2025-03-23 21:19:59 ☺ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADV  
ANCED/scripts  
● o → sudo ./mantenimiento.sh  
Sun Mar 23 21:22:02 CET 2025: Iniciando script de mantenimiento...  
Sun Mar 23 21:22:02 CET 2025: Iniciando limpieza de logs...  
Sun Mar 23 21:22:02 CET 2025: Limpieza de logs completada.  
Sun Mar 23 21:22:02 CET 2025: Verificando estado de servicios...  
Sun Mar 23 21:22:02 CET 2025: Verificación de servicios completada.  
Sun Mar 23 21:22:02 CET 2025: Generando informe de estado en /tmp/status_report_20250323.txt..  
  
Sun Mar 23 21:22:04 CET 2025: Informe de estado generado en /tmp/status_report_20250323.txt.  
Sun Mar 23 21:22:04 CET 2025: Script de mantenimiento completado.  
  
2025-03-23 21:27:04 ☺ Nurias-MacBook-Pro in /tmp  
|o → cat status_report_20250323.txt  
== INFORME DE ESTADO DEL SERVIDOR (Sun Mar 23 21:22:02 CET 2025) ==  
  
== INFORMACIÓN DEL SISTEMA ==  
Hostname: Nurias-MacBook-Pro.local  
IP Addresses:  
192.168.1.129  
  
== USO DE RECURSOS ==  
CPU:  
CPU usage: 7.96% user, 8.54% sys, 83.48% idle  
Memoria:  
Mach Virtual Memory Statistics: (page size of 4096 bytes)  
Pages free: 38478.  
Pages active: 1367615.  
Pages inactive: 1357243.  
Pages speculative: 9723.  
Pages throttled: 0.  
Pages wired down: 888298.  
Pages purgeable: 87985.  
"Translation faults": 343796928.  
Pages copy-on-write: 5513575.  
Pages zero filled: 79134855.  
Pages reactivated: 2988498.  
Pages purged: 1837361.  
File-backed pages: 611098.  
Anonymous pages: 2123483.  
Pages stored in compressor: 1503007.  
Pages occupied by compressor: 531893.  
Decompressions: 7154034.  
Compressions: 10647200.  
Pageins: 4884623.  
Pageouts: 19802.  
Swaps: 5650255.  
Swapouts: 5861926.  
Disco:  
Filesystem Size Used Avail Capacity iused ifree %iused Mounted on  
/dev/disk1s4s1 466Gi 11Gi 341Gi 4% 412k 3.6G 0% /  
devfs 342Ki 0Bi 100% 1.2k 0 100% /dev  
/dev/disk1s2 466Gi 4.2Gi 341Gi 2% 1.7k 3.6G 0% /System/Volumes/Preboot  
/dev/disk1s6 466Gi 2.0Gi 341Gi 1% 2 3.6G 0% /System/Volumes/VM  
/dev/disk1s5 466Gi 67Mi 341Gi 1% 597 3.6G 0% /System/Volumes/Update  
/dev/disk1s1 466Gi 10Gi 341Gi 24% 1.3M 3.6G 0% /System/Volumes/Data  
map auto_home 0Bi 0Bi 0Bi 100% 0 0 - /System/Volumes/Data/home  
/dev/disk1s4 466Gi 11Gi 341Gi 4% 412k 3.6G 0% /System/Volumes/Update/mnt1  
/dev/disk1s3 466Gi 1.2Gi 341Gi 1% 197 3.6G 0% /Volumes/Recovery  
/Users/mmurria/Downloads/Visual Studio Code.app 466Gi 107Gi 342Gi 24% 1.3M 3.6G 0% /private/var/folders/7s/jkpq7pn96dz0j16j48dh3g3m000gn/T/AppTranslocation/9C86EE8A-96E9-4D32-AC73-1FED7DC5F505  
  
== CONEXIONES ACTIVAS ==  
Conexiones establecidas:  
 32  
Conexiones en espera:  
 28  
  
== LOGS RECIENTES ==  
2025-03-23 21:27:11 ☺ Nurias-MacBook-Pro in /tmp
```

Es importante recalcar que **mi sistema operativo es macOs** por lo que hay unos cuantos comandos que para otros sistemas operativos como Linux han de cambiarse:

- Línea 82: `top -l 1 | head -10 | grep "CPU usage" >> $REPORT_FILE`, debe de cambiarse para linux por `top -bn1 | grep "Cpu(s)" >> $REPORT_FILE`
- Línea 84: `vm_stat >> $REPORT_FILE` por `free -m >> $REPORT_FILE`
- Línea 77: `ifconfig | grep "inet " | grep -v 127.0.0.1 | awk '{print $2}' >> $REPORT_FILE` por `ip addr | grep "inet " | grep -v 127.0.0.1 | awk '{print $2}' >> $REPORT_FILE`
- Escribir scripts para automatizar la creación de contenedores o la configuración de la red.

En este caso he optado por la creación de un archivo llamado **despliegue.sh** el cuál se encarga de todo el proceso de configuración, creación y despliegue de los contenedores de forma correcta, comprobando todas las dependencias, directorios y conexiones de red necesarias. Por el mismo motivo que los anteriores, me ceñiré a mostrar el resultado de la ejecución de dicho archivo, ya que, el contenido es algo extenso y se puede consultar en el archivo directamente.



The terminal window shows the execution of the script and a detailed view of the Docker container hierarchy:

```
[+] Running 9/9
✓ Network p1-advanced_red_web      Created          0.1s
✓ Network p1-advanced_red_servicios Create...       0.1s
✓ Container p1-advanced-apache2-1   Started         0.2s
✓ Container network_manager        Started         0.2s
✓ Container p1-advanced-lighttpd1-1 Starte...       0.2s
✓ Container p1-advanced-lighttpd2-1 Starte...       0.2s
✓ Container p1-advanced-nginx2-1   Started         0.2s
✓ Container p1-advanced-nginx1-1   Started         0.2s
✓ Container p1-advanced-apache1-1  Started         0.2s
```

Container	Image	Status	CPU (%)	Memory (MB)	Last Seen	Actions
p1-advanced		Running (8/8)	0.08%	3 minutes ago		
lighttpd2-1		mmnuria-lighttpd:latest	Running	0.01% 8093:80	3 minutes ago	
apache2-1		mmnuria-apache-advanced:lates	Running	0.01% 8089:80	3 minutes ago	
nginx2-1		mmnuria-nginx:latest	Running	0.01% 8091:80	3 minutes ago	
network_manager		alpine:latest	Running	0.01%	3 minutes ago	
apache1-1		mmnuria-apache-advanced:lates	Running	0.01% 8088:80	3 minutes ago	
nginx1-1		mmnuria-nginx:latest	Running	0.01% 8090:80	3 minutes ago	
lighttpd1-1		mmnuria-lighttpd:latest	Running	0.02% 8092:80	3 minutes ago	

Además, esta evidencia corrobora el **funcionamiento correcto de todos los apartados anteriores**, ya que, se han construido y desplegado de forma satisfactoria cada uno de ellos, para confirmar definitivamente su funcionamiento, muestro captura de pantalla del index.php:

- Petición al servidor nginx1:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.20

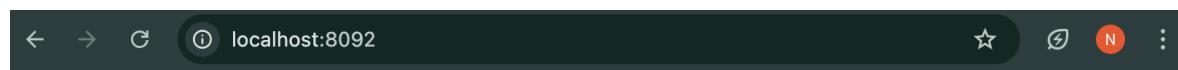
- Petición al servidor nginx2:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.21

- Petición al servidor Lighttpd1:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.30

- Petición al servidor Lighttpd2:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.31

- Petición al servidor Apache1-advanced:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.10

- Petición al servidor Apache2-advances:



Práctica 1 SWAP ADVANCED - Nuria Manzano Mata

La dirección IP del servidor es: 192.168.200.11

Cómo se puede observar, cada petición devuelve una dirección IP diferente, la cuál corresponde a la dirección del servidor que está respondiendo en los diferentes puertos. Si se compara con las direcciones IPs asignadas en la configuración, éstas coinciden.

3.5. A5. Monitoreo y Logging

- Configurar herramientas de monitoreo y logging para rastrear el rendimiento y los eventos de los contenedores.

Éste apartado lo resuelvo en el archivo **monitoreo.sh** que configura herramientas de monitoreo continuo y logging para rastrear el rendimiento y eventos de todos los contenedores Docker.

Funcionamiento:

1. **Registra eventos en un archivo de log** /var/log/container_monitor.log.
2. **Monitorea el uso de CPU** de los contenedores con docker stats, alertando si supera un umbral definido.
3. **Monitorea el uso de memoria**, identificando procesos que consumen más recursos.
4. **Verifica el número de conexiones activas** en los contenedores.
5. **Supervisa el espacio en disco** con docker system df, generando alertas si el uso es alto.
6. **Ejecuta estas comprobaciones en intervalos regulares**, proporcionando información detallada para la administración y optimización de los contenedores.

Como el archivo es bastante extenso, mostraré captura de pantalla del resultado de la ejecución.

```

2025-03-23 21:41:41 ✎ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADVANCED/scripts
o → sudo ./monitoreo.sh
2025-03-23 21:46:12 - ===== Iniciando monitoreo =====
2025-03-23 21:46:12 - Monitoreando uso de CPU por contenedor...
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-nginx1-1' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-nginx2-1' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-lighttpd1-1' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-apache1-1' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-lighttpd2-1' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'network_manager' usa 0% de CPU
2025-03-23 21:46:14 - INFO: Contenedor 'p1-advanced-apache2-1' usa 0% de CPU
2025-03-23 21:46:14 - Monitoreando uso de memoria por contenedor...
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-nginx1-1' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-nginx2-1' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-lighttpd1-1' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-apache1-1' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-lighttpd2-1' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'network_manager' usa 0% de memoria
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-apache2-1' usa 0% de memoria
2025-03-23 21:46:16 - Monitoreando conexiones activas en contenedores...
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-nginx1-1' tiene      0 conexiones activas
2025-03-23 21:46:16 - INFO: Contenedor 'p1-advanced-nginx2-1' tiene      0 conexiones activas
2025-03-23 21:46:17 - INFO: Contenedor 'p1-advanced-lighttpd1-1' tiene      0 conexiones activas
2025-03-23 21:46:17 - INFO: Contenedor 'p1-advanced-apache1-1' tiene      0 conexiones activas
2025-03-23 21:46:17 - INFO: Contenedor 'p1-advanced-lighttpd2-1' tiene      0 conexiones activas
2025-03-23 21:46:17 - INFO: Contenedor 'network_manager' tiene      0 conexiones activas
2025-03-23 21:46:17 - INFO: Contenedor 'p1-advanced-apache2-1' tiene      0 conexiones activas
2025-03-23 21:46:18 - ERROR: No se pudo obtener el uso de disco de Docker
2025-03-23 21:46:18 - Monitoreo completado. Próxima ejecución en 60 segundos.
^C

```

- Utilizar herramientas como htop, netstat, o apache2ctl dentro de los contenedores para monitorear y diagnosticar el estado del servidor.

En este archivo **diagnostico-interno.sh** se configura herramientas dentro del contenedor Docker especificado, para realizar un análisis más profundo.

Funcionamiento:

1. Detecta automáticamente el sistema de paquetes del contenedor (apt, yum, dnf, apk)
2. Verifica e instala herramientas de diagnóstico si no están disponibles (htop, netstat, apache2ctl)
3. Ejecuta diagnósticos usando estas herramientas dentro del contenedor
4. Guarda todos los resultados en un archivo de log

```

2025-03-24 12:52:37 ✘ Nurias-MacBook-Pro in ~/Desktop/UNI/QUINTO/SEGUNDO CUATRI/SWAP/P1-ADVANCED/scripts
○ → ./diagnostico-interno.sh p1-advanced-nginx2-1
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - ===== Iniciando diagnóstico interno del contenedor =====
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - Detectando sistema de paquetes del contenedor...
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - Sistema de paquetes detectado: APT (Debian/Ubuntu)
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - Verificando herramientas disponibles...
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - La herramienta htop está disponible
2025-03-24 12:55:16 - p1-advanced-nginx2-1 - La herramienta netstat está disponible
2025-03-24 12:55:17 - p1-advanced-nginx2-1 - La herramienta apache2ctl/httpd no está disponible
2025-03-24 12:55:17 - p1-advanced-nginx2-1 - Se detectaron herramientas faltantes. Procediendo con la instalación automática.
2025-03-24 12:55:17 - p1-advanced-nginx2-1 - Preparando para instalar herramientas necesarias...
2025-03-24 12:55:17 - p1-advanced-nginx2-1 - Actualizando repositorios APT...
Hit:1 http://deb.debian.org/debian bullseye InRelease
Hit:2 http://deb.debian.org/debian-security bullseye-security InRelease
Hit:3 http://deb.debian.org/debian bullseye-updates InRelease
Reading package lists...
2025-03-24 12:55:19 - p1-advanced-nginx2-1 - Instalando utilidades de Apache...
Reading package lists...
Building dependency tree...
Reading state information...
apache2-utils is already the newest version (2.4.62-1-deb11u2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
2025-03-24 12:55:20 - p1-advanced-nginx2-1 - Utilidades de Apache instaladas.
2025-03-24 12:55:20 - p1-advanced-nginx2-1 - Verificando herramientas disponibles...
2025-03-24 12:55:20 - p1-advanced-nginx2-1 - La herramienta htop está disponible
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - La herramienta netstat está disponible
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - La herramienta apache2ctl/httpd no está disponible
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - Ejecutando diagnóstico general del sistema...
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - Información del sistema:
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - Uso de CPU:
2025-03-24 12:55:21 - p1-advanced-nginx2-1 - Uso de memoria:
2025-03-24 12:55:22 - p1-advanced-nginx2-1 - Espacio en disco:
2025-03-24 12:55:22 - p1-advanced-nginx2-1 - Procesos en ejecución:
2025-03-24 12:55:22 - p1-advanced-nginx2-1 - Diagnóstico general completado
2025-03-24 12:55:22 - p1-advanced-nginx2-1 - Ejecutando diagnóstico con htop...
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Diagnóstico con top completado
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Ejecutando diagnóstico de conexiones con netstat...
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Conexiones establecidas:
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Puertos en escucha:
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Estadísticas de conexiones:
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Diagnóstico con netstat completado
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - =====
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - RESUMEN DEL DIAGNÓSTICO
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - =====
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Contenedor: p1-advanced-nginx2-1
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Sistema de paquetes: apt
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Herramientas disponibles:
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - - htop: SÍ
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - - netstat: SÍ
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - - apache2ctl/httpd: NO
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Archivo de registro completo: /Users/mmnuria/docker_diagnostics/p1-advanced-nginx2-1_diag.log
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - =====
2025-03-24 12:55:23 - p1-advanced-nginx2-1 - Diagnóstico interno completado.

Diagnóstico completado. El archivo de log está en: /Users/mmnuria/docker_diagnostics/p1-advanced-nginx2-1_diag.log

```

El script realiza cinco diagnósticos principales:

- Diagnóstico general (información de sistema, CPU, memoria, disco)
- Diagnóstico detallado de procesos con htop/top
- Análisis de conexiones de red con netstat
- Diagnóstico de Apache si está instalado
- Resumen final con la información recopilada

Todos los resultados se guardan en:

`$HOME/docker_diagnostics/[nombre_contenedor]_diag.log`

3.6. Análisis propuesta IA

Con este apartado voy a tratar de analizar los resultados propuestos por la IA para este segundo apartado de tareas avanzadas. He preguntado a la IA escribiendo directamente el enunciado de las diferentes tareas para que me mostrara paso a paso lo que debe de incluir cada una, [pulsa aquí para acceder al chat](#)

De forma resumida, la IA me ha proporcionado soluciones algo diferentes a lo que había pensado:

1. Personalización del Dockerfile: La propuesta que me sugiere la IA es un enfoque mucho más simple que mi solución ya que, solamente copia archivos de configuración preexistentes, usa configuraciones PHP básicas, configura Apache solamente habilitando AllowOverride y los logs básicos. De forma que, aunque todo esto forma parte de mi solución, he optado por implementaciones extra especialmente en la configuración de Apache (rendimiento + seguridad) y la configuración PHP. Por lo tanto, decido usar la mía.
2. Contenedores de otros servidores web: Lo más importante de este apartado es que la IA no me ha sugerido la necesidad de la integración de cada uno de los servidores con PHP-FPM para el uso de PHP. Además de que, no me ha sugerido tampoco los scripts de inicio de cada servidor web que son tan importantes cuando se trabaja con Docker debido a que, se necesita gestionar múltiples procesos (Nginx/Lighttpd con PHP-FPM en el mismo contenedor) y la secuencia de arranque para que se garantice que PHP-FPM inicie antes que el servidor y así verificar las comunicaciones correctas con los servidores (TCP para Nginx y Socket para Lighttpd). Por lo tanto, decido usar la mía.
3. Gestión avanzada de redes: En este apartado, la IA sólo se ha limitado a sugerirme comandos de Docker simples para crear redes y algún ejemplo de reglas que puedo aplicar, pero, la principal importancia es que no propone una solución integrada. Por lo que, prefiero mi solución en el que creo un contenedor específico con un script “configuracion-redes.sh” que tiene todo el control para automatizar este apartado, aplicando reglas específicas para cada red. Por lo tanto, mi solución es mucho más completa que la que me ha proporcionado la IA.
4. Automatización con scripts: En este caso, la IA también se ha limitado a sugerirme scripts simples en la que me propone automatizarlo con cron, lo cual no conozco. Por todo ello, he optado por usar mi solución ya que, por un lado me parece mucho más ordenada, completa y contiene validaciones necesarias para verificar dependencias, directorios y conexiones.
5. Monitoreo y logging: Cómo sucede en el punto 3, la IA se ha limitado a sugerir comandos básicos de Docker sin ningún tipo de automatización para el monitoreo ni usos de logging que incorporen alertas. Por lo tanto, elijo mi opción en la que

realizo scripts para resolver por un lado el monitoreo y por otro lado un diagnóstico más profundo para un contenedor específico.

En definitiva, la **IA ofrece soluciones básicas que podrían servir como punto de partida**, pero carece de la profundidad, integración y consideraciones prácticas necesarias para un entorno real.

Bibliografía:

- Documentación Docker.
<https://docs.docker.com/desktop/use-desktop/container/>
- Documentación inicio Apache.
<https://httpd.apache.org/docs/2.4/getting-started.html>
- Página inicio PHP <https://www.php.net/manual/es/tutorial.firstpage.php>
- Nginx con Ubuntu. Ejemplo.
<https://linuxize.com/post/how-to-set-up-nginx-server-blocks-on-ubuntu-22-04/>
- Modelo Docker compose
<https://docs.docker.com/compose/intro/compose-application-model/>
- Ejemplos Docker PHP. <https://github.com/topics/docker-php>
- Documentación oficial Lighttpd.
<https://redmine.lighttpd.net/projects/lighttpd/wiki>
- Docker hub imágenes-Nginx. https://hub.docker.com/_/nginx
- Documentación oficial htop. <https://htop.dev/>
- Manual Linux htop. <https://man7.org/linux/man-pages/man1/htop.1.html>
- Documentación oficial net-tools. <https://sourceforge.net/projects/net-tools/>
- Manual Linux netstat. <https://man7.org/linux/man-pages/man8/netstat.8.html>
- Manual Linux apache2ctl.
<https://manpages.debian.org/bookworm/apache2/apache2ctl.8.en.html>