

DAW – WEB SERVER

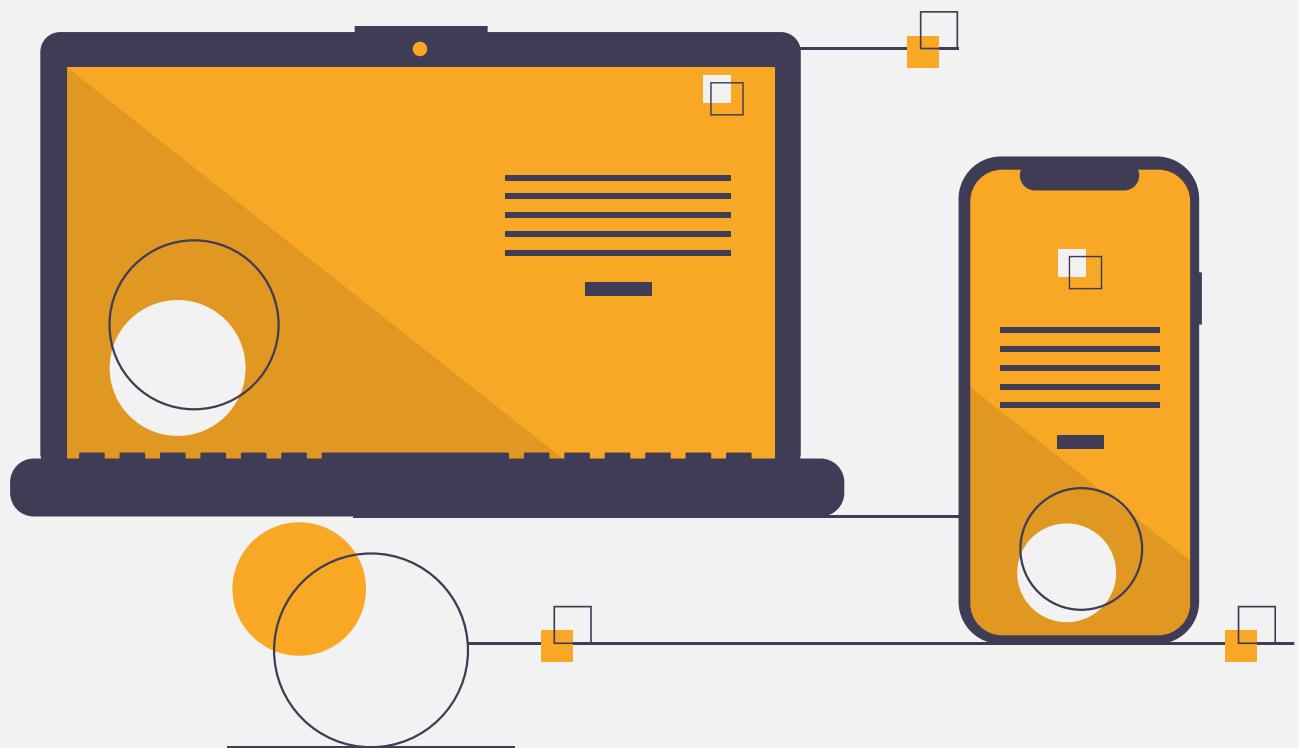
Curso: 2025/26

Profesor:

- M^a José González García

Alumno:

- Diego de la Esperanza de Miguel



Índice

1.	Planteamiento de la arquitectura de red.....	4
a.	Diseño general de la arquitectura.....	4
b.	Selección de tecnologías	4
c.	Diagrama lógico de red.....	5
2.	Instalación y configuración del servidor web en Docker	5
a.	Servidor web seguro (espmig_web)	6
b.	Ficheros de configuración relevantes	6
c.	Automatización del despliegue y borrado de sitios	7
d.	Pruebas de funcionamiento.....	8
3.	Documentación del despliegue y control de versiones	10
a.	Documentación técnica del despliegue.....	10
b.	Control de versiones (Git/GitHub)	14
4.	Conclusiones.....	15
5.	Anexos	16

Tabla de Ilustraciones:

Ilustración 1: Gráfico generado con draw.io (Elaboración propia)	4
Ilustración 2: Diagrama de comunicación entre redes (Elaboración IA).....	5
Ilustración 3: Exposición de puertos y Red de tipo Bridge (Elaboración propia).....	5
Ilustración 4: Docker inspect <nombre de la Red> -> Direcciones IP de los contenedores (Elaboración propia).....	5
Ilustración 5: Ficheros .conf con enlace simbólico en el servidor (Elaboración propia).....	6
Ilustración 6: Barra de búsqueda con https:// (Elaboración propia)	6
Ilustración 7: Ejemplo de fichero dominio.conf con bloques server (Elaboración propia).....	7
Ilustración 8: Ejemplo de fichero de zona para Bind9 (Elaboración propia)	7
Ilustración 9: Prueba de creación de sitio documentacion.doc con el script (Elaboración propia)	8
Ilustración 10: Prueba de conexión con dns al sitio creado (Elaboración propia) ...	8
Ilustración 11: Prueba de acceso desde el navegador (Elaboración propia)	9
Ilustración 12: Prueba de uso de certificados auto firmados (Elaboración propia) .	9
Ilustración 13: Prueba de resolución de nombres nslookup (Elaboración propia)	10
Ilustración 14: Creación del repositorio online (Elaboración propia)	10
Ilustración 15: Inicialización del repositorio local (Elaboración propia)	11
Ilustración 16: Primer commit (Elaboración propia)	11
Ilustración 17: Publicación online (--set-upstream origin main) para crear rama en repositorio vacío, sólo es necesario la primera vez (Elaboración propia)	11
Ilustración 18: Demostración de vista del repositorio (Elaboración propia)	12
Ilustración 19: Agregar .gitignore desde consola (agregar también *.crt) (Elaboración propia).....	12
Ilustración 20: Pasos a seguir para clonar el repositorio (Elaboración propia).....	13
Ilustración 21: Vista al crear el repositorio (la primera vez descargará las imágenes) (Elaboración propia)	13
Ilustración 22: Estructura de directorios (Elaboración propia)	14
Ilustración 23: Página principal del repositorio (Elaboración propia).....	15
Ilustración 24: Historial de los últimos commit (Elaboración propia)	15

1. Planteamiento de la arquitectura de red

a. Diseño general de la arquitectura

Se ha diseñado una infraestructura modular basada en contenedores Docker con el objetivo de aislar servicios, facilitar la escalabilidad y permitir un despliegue reproducible. La arquitectura se compone de los siguientes bloques funcionales:

- Servidor web (Nginx) para la publicación de páginas y gestión de HTTPS.
- Servidor DNS (Bind9) encargado de resolver los dominios internos del proyecto desde el anfitrión Windows.
- Servicios auxiliares destinados a la generación de certificados SSL, así como a la automatización de la creación de nuevos sitios y la consecuente eliminación de estos.

Todos los contenedores se interconectan mediante una red Docker, bajo el driver Bridge lo que garantiza aislamiento, control de tráfico y facilidad de mantenimiento.

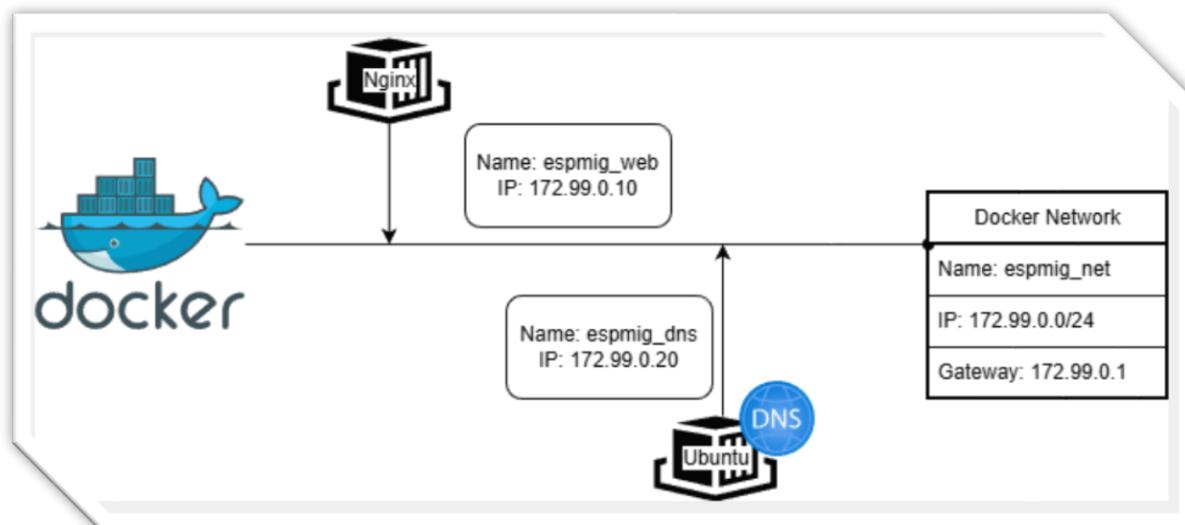


Ilustración 1: Gráfico generado con draw.io (Elaboración propia)

b. Selección de tecnologías

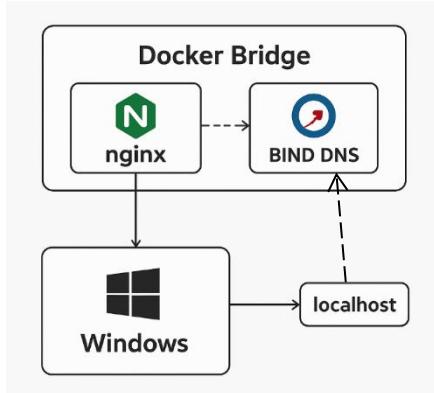
Las tecnologías se han elegido siguiendo criterios de estabilidad, seguridad y facilidad de configuración:

- Imágenes personalizadas, dado su minimalismo y funcionalidad, e imágenes robustas como Ubuntu.
- Nginx por su rendimiento, modularidad y amplio soporte para configuraciones avanzadas de HTTPS.
- Bind9 para la gestión de zonas DNS, dada su robustez y versatilidad.

- Docker Compose para la orquestación del entorno, lo que simplifica despliegue, escalado y mantenimiento.

c. Diagrama lógico de red

La comunicación entre contenedores sigue una estructura clara:



El diagrama representa el flujo de la comunicación en la red, ante una petición de Windows (anfitrión) a localhost, es el servidor dns el encargado de recibirla y dirigir la petición a la ip correcta (servidor web).

Ilustración 2: Diagrama de comunicación entre redes (Elaboración IA)

```

PS C:\Users\diego\OneDrive - Educacyl\GradoSuperior DAW\Docker\DiegoEspMig_DAW\docker-containers> docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
d1a7efefed40        docker-containers-web   "/entrypoint supervi..."   9 hours ago       Up 26 minutes      0.0.0.0:80->80/tcp, 0.0.0.0:843->443/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:53->53/tcp, 0.0.0.0:53->53/udp, 0.0.0.0:53->53/tcp, 0.0.0.0:53->53/udp
PS C:\Users\diego\OneDrive - Educacyl\GradoSuperior DAW\Docker\DiegoEspMig_DAW\docker-containers> docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
78b663ab2c41        bridge              bridge              local
7adeaa952e76        docker-container_espmig_net   bridge              local
ecaeabab1fb85        dmes-docker-lamp_default   bridge              local
26c33dbe7ec6        dmes-examen_default   bridge              local
ebed27910a30        host                host                local
d8183fa90271        none                null               local
  
```

Este cuadro captura la terminal de Windows mostrando el resultado de los comandos docker ps y docker network ls. El comando docker ps muestra un único contenedor llamado 'docker-containers-web' que ejecuta la imagen 'docker-containers-web'. El comando docker network ls muestra una red tipo puente ('bridge') que incluye el host y otros contenedores.

Ilustración 3: Exposición de puertos y Red de tipo Bridge (Elaboración propia)

```

"Containers": {
    "7e6395621cf8718834f14caf9967aa4f9603fc26107711b6157de1b30cbbf78": {
        "Name": "espmig_dns",
        "EndpointID": "abe884680fc94a4ea26d02ccacba8f7719df9301be0842e97d077c0928e4166e",
        "MacAddress": "16:a7:81:d9:09:09",
        "IPv4Address": "172.99.0.20/24",
        "IPv6Address": ""
    },
    "d4e7ef89de849a1b1d2b831d43b7f4f872cd182b51816363a5927efdacd5085e": {
        "Name": "espmig_web",
        "EndpointID": "c7aae7902f503ed0f64406aeeff9ae408c7761b8d5d722de14c881285652ceb20",
        "MacAddress": "8a:73:fa:20:fd:06",
        "IPv4Address": "172.99.0.10/24",
        "IPv6Address": ""
    }
}
  
```

Este cuadro muestra el resultado del comando Docker inspect para las dos interfaces de red ('espmig_dns' y 'espmig_web'). Se detallan sus respectivas direcciones IP (172.99.0.20 y 172.99.0.10), máscaras de subnet (24 bits) y direcciones MAC.

Ilustración 4: Docker inspect <nombre de la Red> -> Direcciones IP de los contenedores (Elaboración propia)

2. Instalación y configuración del servidor web en Docker

La construcción del servidor web se ha realizado mediante un Dockerfile personalizado, apoyado en un docker-compose.yml que gestiona variables, montaje de volúmenes y redes.

Dockerfile: Selecciona la imagen exacta que se va a utilizar en cada contenedor, también ejecuta algunos comandos para mantener el sistema actualizado, instala herramientas

útiles y necesarias en su configuración interna y borra ficheros temporales para no sobrecargar el disco. *(Detallado en el glosario de términos)

Docker-compose.yml: Aplica las configuraciones para construir el compose, éste incorpora ambos contenedores. Se definen qué carpetas locales son transferidas al sistema interno en el arranque a través de enlaces simbólicos, de esta forma tenemos la opción de poder modificarlos desde un editor como VS Code, también se definen nombre, nombre de host, configuraciones de red, puertos expuestos, reinicios programados por si falla, o roles dentro de la red interna, así como la propia red en la que trabajan ambos contenedores. Es el fichero principal de un compose (contenedor de contenedores). *(Detallado en el glosario de términos)

a. Servidor web seguro (espmig_web)

Las características principales son:

- Integración de certificados SSL/TLS auto firmados para habilitar conexiones HTTPS en entornos de desarrollo.
- Redirección automática de HTTP → HTTPS para asegurar el uso de conexiones cifradas.
- Ficheros de configuración independientes para permitir múltiples dominios bajo el mismo servidor, garantizando modularidad.
- Imagen Docker personalizada webdevops/php-nginx
- Servidor nginx por su ligereza y rendimiento.
- Servidor php para aumentar la usabilidad y posibilidades

```
root@servidorNginx:/# ls /etc/nginx/conf.d/
nueva.conf  pagina.conf  proyecto.conf
root@servidorNginx:/#
```

Ilustración 5: Ficheros .conf con enlace simbólico en el servidor (Elaboración propia)

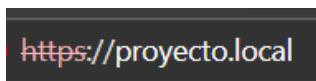


Ilustración 6: Barra de búsqueda con https:// (Elaboración propia)

b. Ficheros de configuración relevantes

Configuración de Nginx (dominio.conf):

- Definición de bloques *server*.
- Inclusión de certificados y claves privadas.
- Redirección a https.

```
# Redirección HTTP a HTTPS
server {
    listen 80;
    server_name proyecto.local www.proyecto.local;
    return 301 https://$host$request_uri;
}

# Servidor HTTPS
server {
    listen 443 ssl;
    server_name proyecto.local www.proyecto.local;

    # Rutas de los certificados SSL
    ssl_certificate /etc/ssl/private/proyecto.crt;
    ssl_certificate_key /etc/ssl/private/proyecto.key;

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

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

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

Ilustración 7: Ejemplo de fichero dominio.conf con bloques server (Elaboración propia)

Archivos de zona DNS (db.dominio.tld y db.172.99.0):

- Registros A y PTR correctamente definidos.
- Estructura limpia y sin duplicidades.

```
$TTL    604800
$ORIGIN proyecto.local.
@      IN      SOA     ns1.proyecto.local. admin.proyecto.local. (
                      2025111505 ; Serial
                      604800      ; Refresh
                      86400       ; Retry
                      2419200    ; Expire
                      604800 )   ; Negative Cache TTL

; Servidor de nombres
@      IN      NS      ns1.proyecto.local.

; Servidor DNS
ns1      IN      A       172.99.0.20

; Acceso desde host Windows (a través del mapeo de puertos de Docker)
@      IN      A       127.0.0.1
www    IN      A       127.0.0.1
```

Ilustración 8: Ejemplo de fichero de zona para Bind9 (Elaboración propia)

c. Automatización del despliegue y borrado de sitios

Se ha implementado un script automatizado que permite:

- Crear un nuevo dominio interno en el DNS.

- Generar el bloque de configuración correspondiente en Nginx.
 - Crear certificados SSL para el nuevo sitio. (Open SSL requerido).
 - Generación de carpeta del sitio web.
 - Reiniciar los servicios.

Este enfoque permite escalar la infraestructura rápidamente y replicarla sin intervención manual.

A su vez, se ha creado otro script que elimina el sitio y la configuración de este tras una doble verificación de borrado.

Ilustración 9: Prueba de creación de sitio documentacion.doc con el script (Elaboración propia)

```
PS C:\WINDOWS\system32> nslookup documentacion.doc 127.0.0.1
Servidor: localhost
Address: 127.0.0.1

Nombre: documentacion.doc
Address: 127.0.0.1

PS C:\WINDOWS\system32>
```

Ilustración 10: Prueba de conexión con dns al sitio creado (Elaboración propia)

d. Pruebas de funcionamiento

Las pruebas realizadas incluyen:

- Acceso a los dominios configurados desde el navegador del anfitrión Windows.

El acceso desde el navegador se recomienda usar Firefox, dado que hace uso de la configuración propia del DNS, otros como Edge hacen uso de cachés en las que si se ha almacenado un error de carga no realizan la petición de nuevo o directamente hacen la petición a servidores que denominan seguros (8.8.8.8 Google), obviando la configuración del adaptador de red

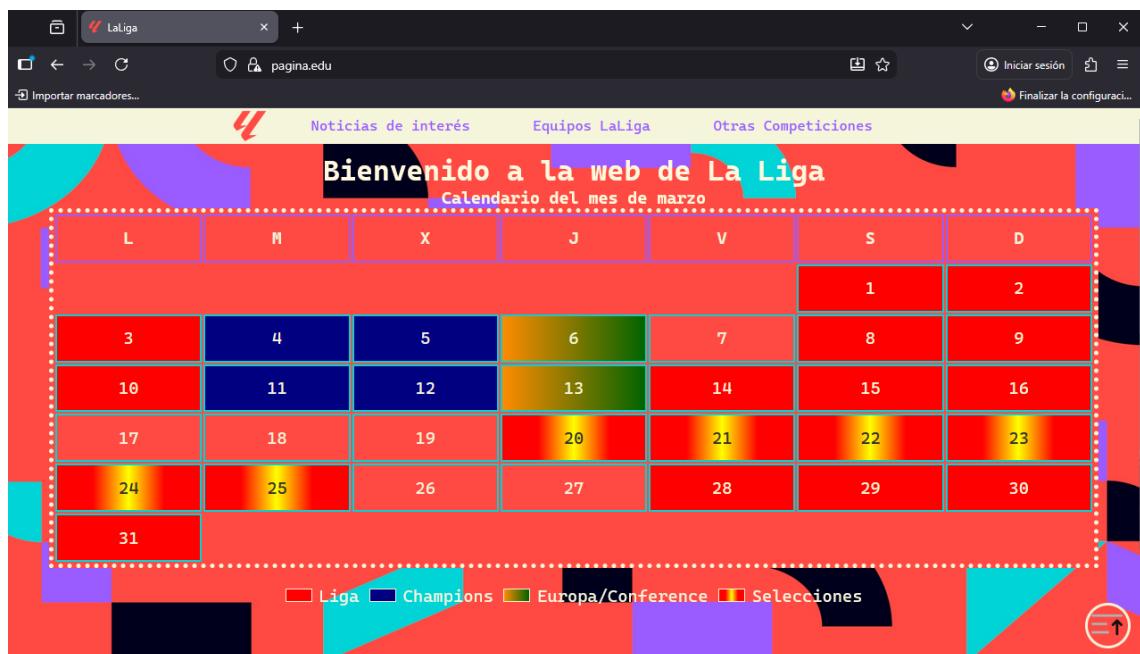


Ilustración 11: Prueba de acceso desde el navegador (Elaboración propia)

- Validación de certificados SSL.

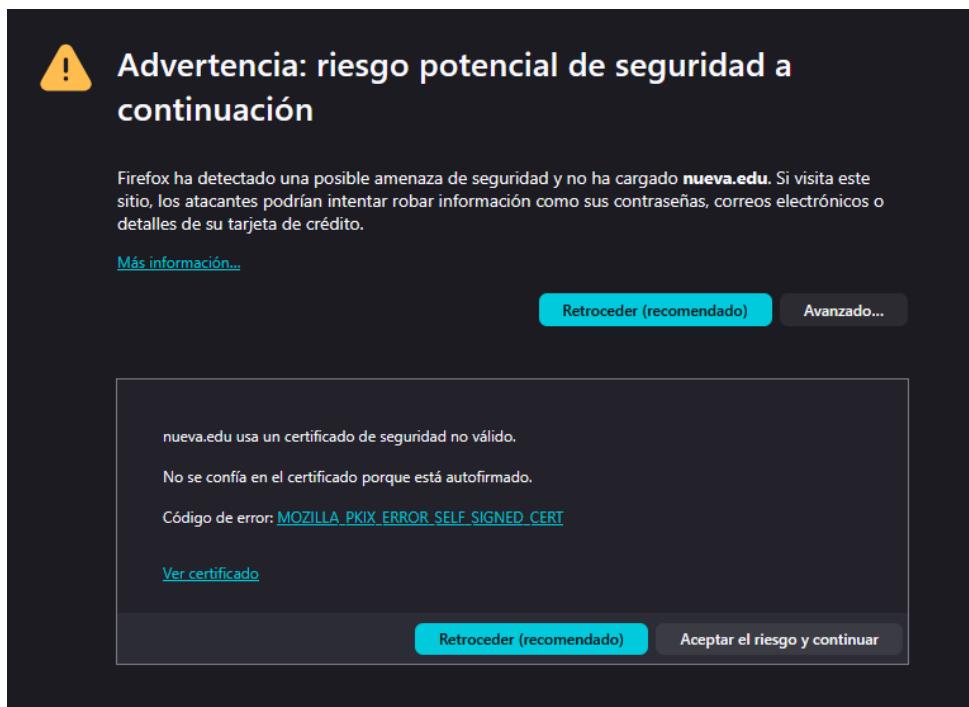


Ilustración 12: Prueba de uso de certificados auto firmados (Elaboración propia)

- Comprobación de resolución DNS mediante nslookup.

```
C:\Users\diego>nslookup documentacion.doc 127.0.0.1
Servidor:  localhost
Address:  127.0.0.1

Nombre:  documentacion.doc
Address:  127.0.0.1

C:\Users\diego>
```

Ilustración 13: Prueba de resolución de nombres nslookup (Elaboración propia)

3. Documentación del despliegue y control de versiones

a. Documentación técnica del despliegue

Se documentan los pasos de creación del repositorio:

1. Creación del repositorio online

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner * **Repository name *** DAW_Docker is available.

Great repository names are short and memorable. How about [ideal-octo-couscous](#)?

Description

53 / 350 characters

2 Configuration

Choose visibility * **Public**

Add README Off READMEs can be used as longer descriptions. [About READMEs](#)

Add .gitignore **No .gitignore**

Add license **No license** Licenses explain how others can use your code. [About licenses](#)

Ilustración 14: Creación del repositorio online (Elaboración propia)

2. Inicialización del repositorio local y primer commit

```
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW
$ git init
Initialized empty Git repository in C:/Users/diego/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW/.git/
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git remote add origin https://github.com/DiegoEspMig/DAW_Docker
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ArquitecturaRedes.drawio
    DiegoEspMig_DAW.docx
    Memoriadíario.txt
    crearRepo.png

nothing added to commit but untracked files present (use "git add" to track)

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ ...
```

Ilustración 15: Inicialización del repositorio local (Elaboración propia)

```
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git add .
warning: in the working copy of 'ArquitecturaRedes.drawio', LF will be replaced by CRLF the next time Git touches it

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git commit -m "Primer commit, primeros ficheros y estructura de carpetas subidas"
[mmain (root-commit) 78f8ebb] Primer commit, primeros ficheros y estructura de carpetas subidas
 5 files changed, 67 insertions(+)
create mode 100644 ArquitecturaRedes.drawio
create mode 100644 DiegoEspMig_DAW.docx
create mode 100644 Memoriadíario.txt
create mode 100644 crearRepo.png
create mode 100644 git1.png

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ ...
```

Ilustración 16: Primer commit (Elaboración propia)

```
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git push --set-upstream origin main
info: please complete authentication in your browser...
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 119.47 KiB | 23.89 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/DiegoEspMig/DAW_Docker
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Ilustración 17: Publicación online (--set-upstream origin main) para crear rama en repositorio vacío, sólo es necesario la primera vez (Elaboración propia)

3. Visualización pública del repositorio

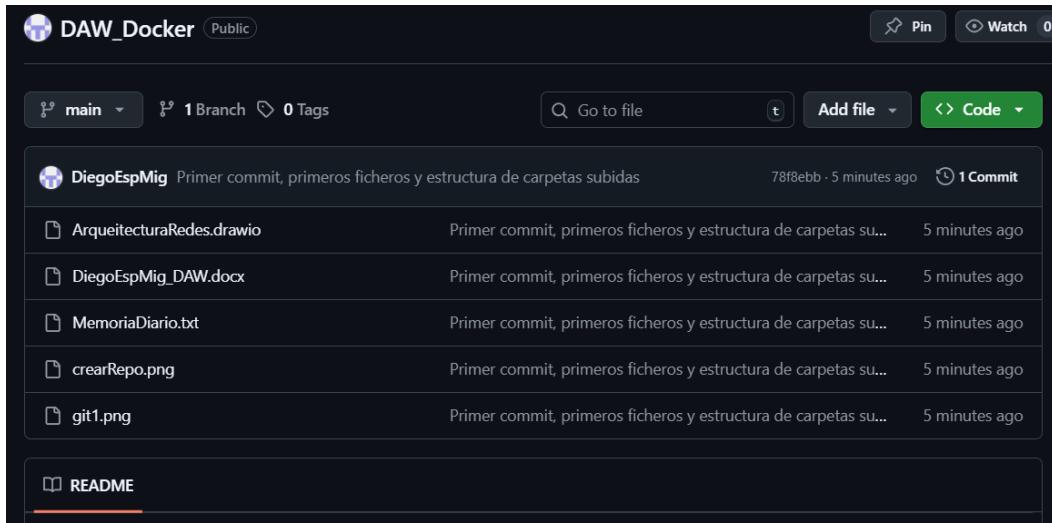


Ilustración 18: Demostración de vista del repositorio (Elaboración propia)

4. Configuración del repositorio para evitar exponer claves públicas y certificados

```
diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ echo "Docker-Container/certs/*.key" >> .gitignore

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git add .gitignore
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git commit -m ".gitignore agregado para evitar exponer claves privadas en certificados"
[main bc23bc0] .gitignore agregado para evitar exponer claves privadas en certificados
 2 files changed, 2 insertions(+), 29 deletions(-)
   delete mode 100644 Docker-Container/certs/proyecto.key

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 500 bytes | 500.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/DiegoEspMig/DAW_Docker
  f3926c0..bc23bc0 main -> main

diego@DiegoEspMig MINGW64 ~/OneDrive - Educacyl/GradoSuperior DAW/Docker/DiegoEspMig_DAW (main)
$
```

Ilustración 19: Agregar .gitignore desde consola (agregar también *.crt) (Elaboración propia)

Se documentan los pasos necesarios para replicar el entorno completo:

1. Clonar el repositorio desde GitHub. [Repositorio GitHub](#)

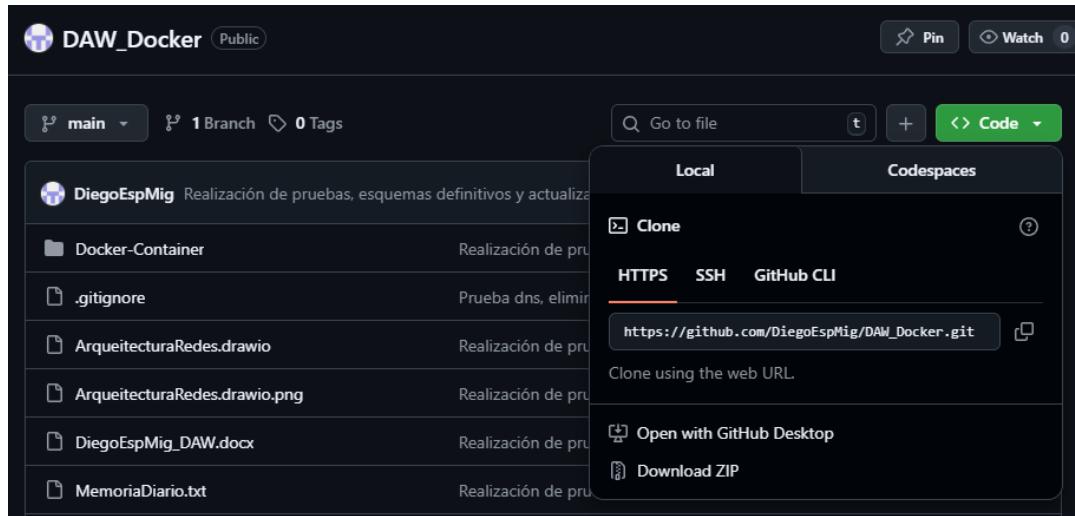


Ilustración 20: Pasos a seguir para clonar el repositorio (Elaboración propia)

2. Levantar los servicios con “`docker-compose up -d --build`” desde la carpeta Docker Container.

```
PS C:\Users\diego\OneDrive - Educacyl\GradoSuperior DAW\Docker\DiegoEspMig_DAW\Docker-Container> docker compose up -d --build
[+] Building 2.4s (15/15) FINISHED
=> [internal] load local bake definitions
=> => reading from stdin 1.28kB
=> [bind9 internal] load build definition from Dockerfile
=> => transferring dockerfile: 453B
=> [web internal] load build definition from Dockerfile
=> => transferring dockerfile: 192B
=> [bind9 internal] load metadata for docker.io/library/ubuntu:latest
=> [web internal] load metadata for docker.io/webdevops/php-nginx:latest
=> [web internal] load .dockerrignore
=> => transferring context: 2B
=> [web 1/2] FROM docker.io/webdevops/php-nginx:latest@sha256:d767e533e0fc17296a51ed36caa4de265fc5b803707e318fc90094538b29a3b4
=> => resolve docker.io/webdevops/php-nginx:latest@sha256:d767e533e0fc17296a51ed36caa4de265fc5b803707e318fc90094538b29a3b4
=> CACHED [web 2/2] RUN apt-get update && apt-get install -y iputils-ping && rm -rf /var/lib/apt/lists/*
=> [web] exporting to image
=> => exporting layers
=> => exporting manifest sha256:608c8afb5fd18a8f9ff88bb41a3681aba78c8099eb082ba55499f89479914061
=> => exporting config sha256:9cd04c2ecfe89898ed21dd10da371b72d3e64ac0f0ace40df3eaa33ea640def3
=> => exporting attestation manifest sha256:850e8de88ff5e28822036c49e57a0bb3d1e4c1ac85e6c1e5da7d05ebbfaf2c695
=> => exporting manifest list sha256:1bfaf2391a2989ebcb208c5e990321437f0ceed96cfdfb8b6448710a16763a95
=> => naming to docker.io/library/docker-container-web:latest
=> => unpacking to docker.io/library/docker-container-web:latest
=> [bind9 internal] load .dockerrignore
=> => transferring context: 2B
=> [bind9 1/2] FROM docker.io/library/ubuntu:latest@sha256:c35e29c9450151419d9448b0fd75374fec4fff364a27f176fb458d472dfc9e54
=> => resolve docker.io/library/ubuntu:latest@sha256:c35e29c9450151419d9448b0fd75374fec4fff364a27f176fb458d472dfc9e54
=> CACHED [bind9 2/2] RUN apt-get update && apt-get upgrade -y && apt-get install -y nano bind9
=> [bind9] exporting to image
=> => exporting layers
=> => exporting manifest sha256:66fc21ee31a322d12e5f8b8996beee24e5ae16fcfc9e99aa2d40fcd303d73a03
=> => exporting config sha256:defdbb5356e117873a3c03c73b22b2b6eb98025af732aa90a0c286b13lef1ac1
=> => exporting attestation manifest sha256:38680fbbaec3e804a519a0cc2066d27e24789899c47e84f383899c4b7265687b
=> => exporting manifest list sha256:447a7e50aa7af985aee61148ac7b1a7de20e54e9c0163697551fcff048237a82a
=> => naming to docker.io/library/docker-container-bind9:latest
=> => unpacking to docker.io/library/docker-container-bind9:latest
=> [web] resolving provenance for metadata file
=> [bind9] resolving provenance for metadata file
[+] Running 5/5
  ✓ docker-container-bind9           Built
  ✓ docker-container-web            Built
  ✓ Network docker-container-espmig_net Created
  ✓ Container espmig_dns           Started
  ✓ Container espmig_web           Started
PS C:\Users\diego\OneDrive - Educacyl\GradoSuperior DAW\Docker\DiegoEspMig_DAW\Docker-Container>
```

Ilustración 21: Vista al crear el repositorio (la primera vez descargará las imágenes) (Elaboración propia)

3. Validar que el DNS está respondiendo. (`nslookup` -> Ilustración 10 y 13)
4. Probar los sitios web en HTTPS. (Ilustración 6, 11 y 12)

Se han incluido explicaciones detalladas de cada fichero relevante:

- Dockerfile

- docker-compose.yml
- dominio.conf
- Archivos de zona DNS
- Script de automatización

```
PS C:\Users\diego\OneDrive - Educacyl\Grado Superior DAW\Docker\DiegoEspMig_DAW> tree /f
Listado de rutas de carpetas
El número de serie del volumen es A058-1EEC
C:.
├── .gitignore
├── ArquitecturaRedes.drawio
├── DiegoEspMig_DAW.docx
└── MemoriaDiario.txt

└── Docker-Container
    ├── borrar-sitio.ps1
    ├── crear-sitio.ps1
    ├── docker-compose.yml
    └── readme.md

    └── certs
        ├── documentacion.crt
        ├── documentacion.key
        ├── nueva.crt
        ├── nueva.key
        ├── pagina.crt
        ├── pagina.key
        ├── proyecto.crt
        └── proyecto.key

    └── conf
        └── dns
            ├── Dockerfile
            ├── named.conf.local
            └── named.conf.options

            └── zones
                ├── db.172.99.0
                ├── db.documentacion.doc
                ├── db.nueva.edu
                ├── db.pagina.edu
                └── db.proyecto.local

        └── nginx
            ├── Dockerfile
            ├── documentacion.conf
            ├── nueva.conf
            ├── pagina.conf
            └── proyecto.conf

    └── www
        └── documentacion
            └── index.php

        └── nueva
            └── index.html
```

Ilustración 22: Estructura de directorios (Elaboración propia)

b. Control de versiones (Git/GitHub)

El proyecto está versionado de forma organizada:

- Commits frecuentes y descriptivos.
- Documentación integrada (readme.md).

DAW_Docker Public

main · 1 Branch · 0 Tags

Go to file Add file Code

DiegoEspMig Avance de la documentacion y agregación de script de automatización d4da597 · 2 minutes ago 16 Commits

Docker-Container Avance de la documentacion y agregación de script de auto... 2 minutes ago

.gitignore Prueba dns, eliminacion de 2a web para simplificar 3 days ago

ArquitecturaRedes.drawio Avance de la documentacion y agregación de script de auto... 2 minutes ago

DiegoEspMig_DAW.docx Avance de la documentacion y agregación de script de auto... 2 minutes ago

MemoriaDiario.txt Realización de pruebas, esquemas definitivos y actualización... 18 hours ago

README

Ilustración 23: Página principal del repositorio (Elaboración propia)

Commits

main · All users · All time

Commits on Nov 17, 2025

Avance de la documentacion y agregación de script de automatización d4da597 · 2 minutes ago

Commits on Nov 16, 2025

Realización de pruebas, esquemas definitivos y actualización de la documentacion 69fa3f6 · 18 hours ago

dns funcional + archivos nuevos que git commit -am no agregó 72b5555 · yesterday

dns funcional d2f0da7 · yesterday

Commits on Nov 15, 2025

Cambio de dns a Bind 18109a7 · 2 days ago

Commits on Nov 14, 2025

Elimino los .crt de github, no en local, que se han agregado a gitignore para no exponer esta informacion 25568a6 · 3 days ago

Prueba dns, eliminacion de 2a web para simplificar 5ccdf0a · 3 days ago

Ilustración 24: Historial de los últimos commit (Elaboración propia)

4. Conclusiones

- Se ha desplegado una infraestructura modular, segura y completamente automatizada utilizando Docker.
- La combinación de (Nginx + PHP) + Bind9 permite gestionar múltiples dominios internos con soporte HTTPS.
- Los scripts desarrollados facilitan la escalabilidad del entorno, reduciendo la intervención manual.

- GitHub proporciona trazabilidad, control de cambios y facilidad para colaborar o evolucionar el proyecto.

5. Anexos

Incluyen:

- Directorio del contenedor.
- Script de automatización.
- Glosario de términos.
- Diagrama de arquitectura de red.
- Memoria del seguimiento de realización del proyecto.
- Documentación (fichero Word y readme.md).