Guía de Despliegue Completo para RAMON en AWS

IMPORTANTE: Rutas a nuestros backups completos

Componente	Ubicación backup	Tamaño	Función
Base de datos	`C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-db-imagen-completa.tar.gz`	149.54 KB	Imagen completa y autónoma de PostgreSQL 17 con todas las tablas y datos migrados
Backend	`C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-api-imagen-completa.tar.gz`	134 MB	Imagen completa y autónoma del backend (FastAPI) con toda la aplicación configurada
Frontend	*Pendiente creación*	-	-
SQL Dump	`C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-imperi-dump.sql`	Variable	Copia completa de la estructura y datos SQL para restauración independiente

Componente	Ubicación backup	Tamaño	Función
Env Config	`C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-backend-env.txt`	< 1 KB	Configuración y variables de entorno del backend

ATENCIÓN: Aclaración importante sobre las imágenes

Los backups de base de datos y backend **contienen absolutamente todo lo necesario** para desplegar estos componentes. No se necesita descargar ni instalar software por separado, ya que:

Sobre el backup de base de datos `masclet-db-imagen-completa.tar.gz`

- Es una imagen Docker completa y autónoma
- Incluye PostgreSQL 17 y todas sus dependencias
- Contiene todas las tablas y datos ya migrados
- Está lista para funcionar solo con cargarla y arrancarla

Sobre el backup de backend `masclet-api-imagen-completa.tar.gz`

- Es una imagen Docker completa y autónoma
- Incluye la API de FastAPI con toda su configuración
- Contiene todas las dependencias Python y bibliotecas necesarias
- Está lista para funcionar solo con cargarla y arrancarla

IMPORTANTE: Configuración crítica para que todo funcione

Para que el sistema funcione correctamente, es FUNDAMENTAL asegurarse de que:

1. La base de datos PostgreSQL debe tener:
    - Una base de datos llamada `masclet_imperi`
    - Un usuario `admin` con contraseña `admin123`
    - El usuario `admin` debe tener permisos para la base de datos `masclet_imperi`
2. El backend está configurado para conectarse mediante:
    - Conexión: `postgres://admin:admin123@masclet-db:5432/masclet_imperi`

Si estos requisitos no se cumplen, el backend dará error de conexión a la base de datos.

Aunque aparentemente los tamaños comprimidos parecen pequeños comparados con el tamaño en el servidor, esto se debe a la excelente compresión que realiza `gzip`, sin pérdida de información.

Objetivo

Este documento proporciona una guía paso a paso para desplegar la aplicación Masclet Imperi en AWS, con instrucciones específicas para garantizar un despliegue sin errores ni complicaciones.

Obtener la clave AWS .pem para conectarse al servidor

1. **Crear par de claves en AWS Console**:
    - Accede a la consola AWS > EC2 > "Key Pairs" > "Create key pair"
    - Nombra la clave (ej: "masclet-imperi-key") y selecciona formato .pem
    - Descárgala automáticamente (solo se podrá descargar UNA vez)
2. **Guardar la clave en un lugar seguro**:
    - Guárdala en una carpeta segura (ej: `C:\Proyectos\AWS\`)
    - NUNCA compartas esta clave - contiene acceso completo al servidor

3. **Configurar permisos de la clave**:
    - En Windows: Click derecho > Propiedades > Seguridad > Avanzado > Solo tu usuario debe tener acceso
    - En Linux/Mac: `chmod 400 tu-clave-aws.pem`

1. Requisitos previos

- Tener acceso a la instancia AWS con la clave `.pem` correcta
- Tener Docker y Docker Compose instalados en la instancia
- Tener un backup actualizado de la base de datos

2. Proceso de despliegue garantizado

2.1. Preparación de la instancia AWS

```
# Conectarse a la instancia
ssh -i "ruta-a-la-clave.pem" ec2-user@54.217.31.124

# Crear estructura de carpetas
mkdir -p ~/masclet-imperi/{db,backend,frontend,config,logs,backups,exito}

# Crear directorio específico para backups automáticos con permisos correctos
sudo mkdir -p /var/backups/masclet-imperi
sudo chmod 777 /var/backups/masclet-imperi

# Instalar Docker y Docker Compose (si no están ya instalados)
sudo yum update -y
sudo yum install -y docker
sudo service docker start
sudo usermod -a -G docker ec2-user
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose

# Reconectar para aplicar cambios de grupo
exit
# Volver a conectar
ssh -i "ruta-a-la-clave.pem" ec2-user@54.217.31.124
```

> **NOTA**: El directorio `/var/backups/masclet-imperi` es CRÍTICO para permitir las copias de seguridad automáticas del sistema.

2.2. Configuración de la base de datos

1. Crear el archivo de configuración en `~/masclet-imperi/config/db.env`:

```
POSTGRES_USER=admin
POSTGRES_PASSWORD=admin123
POSTGRES_DB=masclet_imperi
PGDATA=/var/lib/postgresql/data/pgdata
```

2. Crear el archivo docker-compose en `~/masclet-imperi/docker-compose.yml`:

```
version: '3.8'

services:
  db:
    container_name: masclet-db
    image: postgres:17
    restart: always
    env_file:
      - config/db.env
    ports:
      - "5432:5432"
    volumes:
      - ~/masclet-imperi/db/data:/var/lib/postgresql/data
      - ~/masclet-imperi/backups:/backups
    environment:
      - TZ=Europe/Madrid
    networks:
      - masclet-network

  backend:
    container_name: masclet-backend
    image: masclet-imperi-api:latest
    build:
      context: ./backend
      dockerfile: Dockerfile
    restart: always
    env_file:
      - config/backend.env
    ports:
      - "8000:8000"
    depends_on:
      - db
    volumes:
      - ~/masclet-imperi/logs:/app/logs
    networks:
      - masclet-network

  frontend:
    container_name: masclet-frontend
    image: masclet-imperi-frontend:latest
```

```
    build:
      context: ./frontend
      dockerfile: Dockerfile
    restart: always
    ports:
      - "80:80"
    depends_on:
      - backend
    networks:
      - masclet-network

networks:
  masclet-network:
    name: masclet-network
    driver: bridge
```

3. Crear el archivo de configuración del backend en `~/masclet-imperi/config/backend.env`:

```
DB_USER=admin
DB_PASSWORD=admin123
DB_NAME=masclet_imperi
DB_HOST=masclet-db
DB_PORT=5432
```

2.3. Despliegue de contenedores mediante imágenes completas (MÉTODO RECOMENDADO)

```
# 0. Transferir los archivos de imágenes (ejecutar desde Windows)
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem"
"C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-db-imagen-completa.tar.gz"
ec2-user@54.217.31.124:/home/ec2-user/
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem"
"C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-api-imagen-completa.tar.gz"
ec2-user@54.217.31.124:/home/ec2-user/

# 1. Cargar las imágenes Docker (en el servidor)
docker load < /home/ec2-user/masclet-db-imagen-completa.tar.gz
docker load < /home/ec2-user/masclet-api-imagen-completa.tar.gz

# 2. Iniciar el contenedor de la base de datos (IMPORTANTE: usar el nombre correcto
de la imagen)
# El nombre de la imagen puede ser algo como masclet-db-backup-20250605_125012
docker images
# Usar el nombre de imagen que aparezca en el listado
docker run -d --name masclet-db -p 5432:5432 NOMBRE_DE_TU_IMAGEN_DE_BD
```

```
# 3. CRÍTICO: Verificar que el contenedor de base de datos tiene configurados
correctamente:
# - La base de datos masclet_imperi
# - El usuario admin con contraseña admin123
docker exec masclet-db psql -U postgres -c '\l'
docker exec masclet-db psql -U postgres -c '\du'

# 4. Si falta la base de datos o el usuario, crearlos (OBLIGATORIO)
docker exec masclet-db psql -U postgres -c "CREATE DATABASE masclet_imperi;"
docker exec masclet-db psql -U postgres -c "CREATE USER admin WITH PASSWORD
'admin123' SUPERUSER;"
docker exec masclet-db psql -U postgres -c "GRANT ALL PRIVILEGES ON DATABASE
masclet_imperi TO admin;"

# 5. Iniciar el contenedor del backend (IMPORTANTE: usar el nombre correcto de la
imagen)
# El nombre de la imagen puede ser algo como masclet-api-backup-20250605_125012
docker images
docker run -d --name masclet-backend -p 8000:8000 --link masclet-db:masclet-db
NOMBRE_DE_TU_IMAGEN_DE_BACKEND

# 6. Verificar que los contenedores están funcionando
docker ps

# 7. Verificar logs del backend para confirmar conexión exitosa a la BD
docker logs masclet-backend
```

2.4. Despliegue y restauración alternativo usando SQL Dump (SOLO SI ES NECESARIO)

```
# 1. Iniciar una base de datos PostgreSQL vacía
docker run -d --name masclet-db -p 5432:5432 -e POSTGRES_PASSWORD=postgres
postgres:13

# 2. Transferir el archivo de backup SQL desde local (ejecutar desde Windows)
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem"
"C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-imperi-dump.sql" ec2-
user@54.217.31.124:/home/ec2-user/masclet-backup.sql

# 3. Crear la base de datos y el usuario necesarios
docker exec masclet-db psql -U postgres -c "CREATE DATABASE masclet_imperi;"
docker exec masclet-db psql -U postgres -c "CREATE USER admin WITH PASSWORD
'admin123' SUPERUSER;"
docker exec masclet-db psql -U postgres -c "GRANT ALL PRIVILEGES ON DATABASE
masclet_imperi TO admin;"

# 4. Restaurar el backup en el contenedor
cat /home/ec2-user/masclet-backup.sql | docker exec -i masclet-db psql -U postgres
-d masclet_imperi
```

```
# 5. Verificar que la restauración fue exitosa
docker exec masclet-db psql -U admin -d masclet_imperi -c '\dt'
docker exec masclet-db psql -U admin -d masclet_imperi -c 'SELECT COUNT(*) FROM
animal;'
docker exec masclet-db psql -U admin -d masclet_imperi -c 'SELECT COUNT(*) FROM
part;'
docker exec masclet-db psql -U admin -d masclet_imperi -c 'SELECT COUNT(*) FROM
users;'
```

```
### 2.4. Despliegue del backend y frontend

```bash
# 1. Transferir código fuente del backend y frontend
# (Ejecutar esto desde la máquina local)
# Comprimir código
tar -czvf masclet-code.tar.gz backend frontend

# Transferir a AWS
scp -i "ruta-a-la-clave.pem" masclet-code.tar.gz ec2-user@IP-DE-LA-INSTANCIA:~/

# 2. En la instancia AWS, descomprimir el código
cd ~/masclet-imperi
tar -xzvf ~/masclet-code.tar.gz

# 3. Desplegar todos los servicios
docker-compose up -d

# 4. Verificar que todos los contenedores están funcionando
docker ps
```

3. Comandos para crear backups completos de todos los componentes

3.1. Verificar nombres y estado de los contenedores actuales

```
# Ejecutar en SSH para ver todos los contenedores en ejecución
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker ps -a"
```

Salida esperada (los nombres pueden variar):
```

```
CONTAINER ID   IMAGE                          COMMAND                  STATUS
PORTS                    NAMES
8e0b93ef826c   masclet-backend-masclet-api   "uvicorn app.main:ap..."   Up
(healthy) 0.0.0.0:8000->8000/tcp   masclet-api
cedae60121f8   postgres:17                    "docker-entrypoint.s..."   Up
0.0.0.0:5432->5432/tcp   masclet-db
```

3.2. Crear backup de la base de datos (PostgreSQL)

```
# 1. Crear imagen a partir del contenedor en ejecución
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker commit masclet-db masclet-db-imagen-completa"

# 2. Guardar la imagen como archivo comprimido
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker save masclet-db-imagen-completa | gzip > masclet-db-
imagen-completa.tar.gz"

# 3. Verificar el tamaño del archivo
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "ls -lh masclet-db-imagen-completa.tar.gz"

# 4. Transferir el archivo a local
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119:masclet-db-imagen-completa.tar.gz
"C:\Proyectos\AWS\contenedores despliegue RAMON\"
```

3.3. Crear backup del backend (FastAPI)

```
# 1. Crear imagen a partir del contenedor en ejecución
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker commit masclet-api masclet-api-imagen-completa"

# 2. Guardar la imagen como archivo comprimido
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker save masclet-api-imagen-completa | gzip > masclet-api-
imagen-completa.tar.gz"

# 3. Verificar el tamaño del archivo
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "ls -lh masclet-api-imagen-completa.tar.gz"

# 4. Transferir el archivo a local
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
```

```
user@108.129.139.119:masclet-api-imagen-completa.tar.gz
"C:\Proyectos\AWS\contenedores despliegue RAMON\"
```

3.4. Crear backup del frontend (cuando esté disponible)

```
# 1. Crear imagen a partir del contenedor en ejecución
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker commit masclet-frontend masclet-frontend-imagen-
completa"

# 2. Guardar la imagen como archivo comprimido
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker save masclet-frontend-imagen-completa | gzip >
masclet-frontend-imagen-completa.tar.gz"

# 3. Verificar el tamaño del archivo
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "ls -lh masclet-frontend-imagen-completa.tar.gz"

# 4. Transferir el archivo a local
scp -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119:masclet-frontend-imagen-completa.tar.gz
"C:\Proyectos\AWS\contenedores despliegue RAMON\"
```

3.5. Verificar espacio disponible en el servidor AWS

```
# Ver uso de disco y espacio disponible
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "df -h"

# Ver tamaño de directorios principales
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "du -h --max-depth=1 /home/ec2-user"

# Ver imágenes Docker y su tamaño
ssh -i "C:\Proyectos\primeros proyectos\AWS\masclet-imperi-key.pem" ec2-
user@108.129.139.119 "docker images"
```

4. Restauración rápida de contenedores en una nueva instancia

Para restaurar todo el sistema rápidamente en una nueva instancia:

4.1 Despliegue de la base de datos

Paso 1: Transferir el archivo de imagen Docker al servidor AWS

```
scp -i "tu-clave-aws.pem" "C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-db-imagen-completa.tar.gz" ec2-user@tu-ip-aws:/home/ec2-user/
```

Paso 2: Cargar la imagen Docker

```
docker load < masclet-db-imagen-completa.tar.gz
```

Paso 3: Verificar que la imagen se ha cargado correctamente

```
docker images | grep masclet-db-imagen-completa
```

Paso 4: Ejecutar el contenedor

```
docker run -d --name masclet-db -p 5432:5432 --restart always masclet-db-imagen-completa
```

4.2 Despliegue del backend

Paso 1: Transferir el archivo de imagen Docker al servidor AWS

```
scp -i "tu-clave-aws.pem" "C:\Proyectos\AWS\contenedores despliegue RAMON\masclet-api-imagen-completa.tar.gz" ec2-user@tu-ip-aws:/home/ec2-user/
```

Paso 2: Cargar la imagen Docker

```
docker load < masclet-api-imagen-completa.tar.gz
```

Paso 3: Verificar que la imagen se ha cargado correctamente

```
docker images | grep masclet-api-imagen-completa
```

Paso 4: Crear una red Docker si no existe

```
docker network create masclet-network || true
```

Paso 5: Ejecutar el contenedor del backend

```
docker run -d --name masclet-api -p 8000:8000 --network masclet-network --restart
always masclet-api-imagen-completa
```

```
docker images
```

Deberías ver la imagen `masclet-db-imagen-completa` en la lista.

Paso 4: Iniciar un contenedor desde la imagen

```
docker run -d --name masclet-db -p 5432:5432 -v masclet-db-
data:/var/lib/postgresql/data masclet-db-imagen-completa
```

IMPORTANTE: No es necesario especificar variables de entorno como `POSTGRES_USER`, `POSTGRES_PASSWORD` o `POSTGRES_DB` porque la imagen ya las contiene configuradas exactamente como estaban en el servidor original.

Paso 5: Crear un archivo `docker-compose.yml` para usar la imagen restaurada

```
version: '3.8'

services:
  db:
    container_name: masclet-db
    image: masclet-db-imagen-completa
    restart: always
    ports:
      - "5432:5432"
    volumes:
      - ~/masclet-imperi/backups:/backups
    networks:
      - masclet-network

  backend:
    container_name: masclet-backend
    image: masclet-backend-imagen-completa
    restart: always
    ports:
      - "8000:8000"
    depends_on:
      - db
    volumes:
      - ~/masclet-imperi/logs:/app/logs
    networks:
      - masclet-network

  frontend:
    container_name: masclet-frontend
    image: masclet-frontend-imagen-completa
    restart: always
    ports:
      - "80:80"
    depends_on:
      - backend
    networks:
      - masclet-network

networks:
  masclet-network:
    name: masclet-network
    driver: bridge
EOF

# 4. Iniciar todos los servicios
docker-compose -f docker-compose-prebuilt.yml up -d

# 5. Verificar que todos los contenedores están funcionando
docker ps
```

Este método garantiza un despliegue inmediato sin necesidad de scripts complejos, compilaciones o configuraciones adicionales. Todo está preconfigurado en las imágenes.