AI-03963 Fix Docmost Docker Deployment

Owner	Ⓡ Rigel Arcayan
Tags	Project
Created time	June 22, 2026 12:39 AM

Convert the following docker-compose.yml but configure it with a separate postgres-shared configuration which running in dockerlab-network bridge. Also, provide steps on how to configure the docmost db in pgAdmin to create the db and user. Also, I have an NPM proxy (dockerlab-network) which I need instructions on to setup the proxy host

Host-postgres-shared
bridge network-dockerlab-network
user-postgres
password-SecurePassword1
domain-blindcaveman.duckdns.org
path-/home/rarcayan/dockerlab/apps

```
services:
  docmost:
    image: docmost/docmost:latest
    depends_on:
      - db
      - redis
    environment:
      APP_URL: "http://localhost:3000"
      APP_SECRET: "REPLACE_WITH_LONG_SECRET"
      DATABASE_URL:
"postgresql://docmost:STRONG_DB_PASSWORD@db:5432/docmost"
      REDIS_URL: "redis://redis:6379"
    ports:
      - "3000:3000"
```

```
    restart: unless-stopped
    volumes:
      - docmost:/app/data/storage

db:
    image: postgres:18
    environment:
      POSTGRES_DB: docmost
      POSTGRES_USER: docmost
      POSTGRES_PASSWORD: STRONG_DB_PASSWORD
    restart: unless-stopped
    volumes:
      - db_data:/var/lib/postgresql

redis:
    image: redis:8
    command: ["redis-server", "--appendonly", "yes", "--maxmemory-policy",
"noeviction"]
    restart: unless-stopped
    volumes:
      - redis_data:/data

volumes:
  docmost:
  db_data:
  redis_data:
```

Here is everything you need to transition Docmost to your shared PostgreSQL database, hook it into your custom `dockerlab-network`, and get your Nginx Proxy Manager (NPM) and pgAdmin configurations sorted.

1. The Updated `docker-compose.yml`

Since PostgreSQL is now running separately as a shared service (`postgres-shared`) on the `dockerlab-network`, we can completely remove the local `db` service from this file.

We will define `dockerlab-network` as an **external network** so Docmost can communicate with both your shared database and your NPM proxy.

YAML

```
version: '3.8'

services:
  docmost:
    image: docmost/docmost:latest
    depends_on:
      - redis
    environment:
      # Updated to your actual domain via HTTPS (assuming NPM handles SSL)
      APP_URL: "https://blindcaveman.duckdns.org"
      APP_SECRET: "REPLACE_WITH_LONG_SECRET" # Make sure to generate a secure random stri
ng here
      # Connecting to the shared postgres host on the same bridge network
      DATABASE_URL: "postgresql://docmost:STRONG_DB_PASSWORD@postgres-shared:5432/docmos
t"
      REDIS_URL: "redis://redis:6379"
    expose:
      - "3000" # Exposed to the dockerlab-network for NPM; standard ports mapping removed
for security
    restart: unless-stopped
    volumes:
      - /home/rarcayan/dockerlab/apps/docmost:/app/data/storage
    networks:
      - dockerlab-network

  redis:
    image: redis:8
    command: ["redis-server", "--appendonly", "yes", "--maxmemory-policy", "noeviction"]
    restart: unless-stopped
    volumes:
      - /home/rarcayan/dockerlab/apps/docmost/redis_data:/data
    networks:
      - dockerlab-network

networks:
  dockerlab-network:
    external: true
```

> 💡 **Note:** I changed your `volumes` to bind-mounts pointing to your requested path (`/home/rarcayan/dockerlab/apps/...`) so everything stays neatly organized in your host directory.

2. pgAdmin Configuration (Database & User Setup)

Since your `postgres-shared` container and pgAdmin are likely on the same `dockerlab-network`, here is how to provision the new database.

Step 1: Connect to your Shared Postgres Instance

1. Open pgAdmin and right-click **Servers** → **Register** → **Server...**
2. In the **General** tab, name it `Shared Postgres`.
3. In the **Connection** tab:
   - **Host name/address:** `postgres-shared` (or the IP of the container/host)
   - **Maintenance database:** `postgres`
   - **Username:** `postgres`
   - **Password:** `SecurePassword1`
4. Click **Save**.

Step 2: Create the Docmost User (Login/Group Role)

1. Expand your new server, right-click **Login/Group Roles** → **Create** → **Login/Group Role...**
2. **General Tab:** Set *Role name* to `docmost`.
3. **Definition Tab:** Set *Password* to `STRONG_DB_PASSWORD` (match what you put in the compose file).
4. **Privileges Tab:** Ensure **Can login?** is toggled to **Yes**.
5. Click **Save**.

Step 3: Create the Docmost Database

1. Right-click **Databases** → **Create** → **Database...**
2. **General Tab:**
   - **Database:** `docmost`
   - **Owner:** Select `docmost` from the dropdown.

3. Click **Save**.

3. Nginx Proxy Manager (NPM) Setup

Because Docmost and NPM are sharing the `dockerlab-network`, NPM can route traffic directly to the container using its service name.

1. Log into your **Nginx Proxy Manager** admin dashboard.
2. Go to **Hosts** → **Proxy Hosts** → **Add Proxy Host**.
3. Configure the **Details** tab:
    - **Domain Names:** `blindcaveman.duckdns.org`
    - **Scheme:** `http`
    - **Forward Hostname / IP:** `docmost` *(Docker's internal DNS handles this)*
    - **Forward Port:** `3000`
    - **Cache Assets:** Optional (Enabled recommended)
    - **Block Common Exploits:** Enabled
    - **Websockets Support: Enabled** *(Crucial for Docmost collaborative real-time editing)*
4. Configure the **SSL** tab:
    - **SSL Certificate:** Select your Let's Encrypt certificate (or Request a new one).
    - **Force SSL:** Enabled
    - **HTTP/2 Support:** Enabled
5. Click **Save**.