

# Docker Compose ??

NOTE: ?????? docker compose ?

NOTE: ??????????docker-compose.yml ?????????????????????? docker-compose ???  
container??

??????????????

```
# For all services
docker-compose up -d

# For specified service
docker-compose up -d <service-name>
```

Build the image of the service

```
docker-compose build <service-name>
docker-compose up --rebuild <service-name>
```

????????????

```
docker-compose ps
```

???????????????????? container

```
docker-compose stop
docker-compose start
```

“ ?? container ???

???????????????????? container

# For all services

docker-compose down

# For a specified service

docker-compose stop <service-name>

docker-compose rm -f <service-name>

#### “ NOTE:

- ?????? docker-compose.yml ?????? down ?????? up -d ??????

- ?????????????????? docker-compose up -d

??????????????

docker-compose logs

????????????????

docker-compose exec <service-name> sh -c "pwd"

????? --env-file

docker compose --env-file .env up -d

## docker-compose.yml

environment + healthcheck

services:

db:

image: pgvector/pgvector:0.6.2-pg15

restart: always

ports:

- '5432:5432'

environment:

POSTGRES\_USER: talkdai

POSTGRES\_PASSWORD: talkdai

POSTGRES\_DB: talkdai

```

volumes:
  - ./etc/db-extensions.sql:/docker-entrypoint-initdb.d/db-extensions.sql
healthcheck:
  test: ["CMD", "pg_isready", "-d", "talkdai", "-U", "talkdai"]
  interval: 10s
  timeout: 5s
  retries: 5
dialog:
  image: ghcr.io/talkdai/dialog:latest
  volumes:
    - ./data:/app/data/
  ports:
    - '8000:8000'
  depends_on:
    db:
      condition: service_healthy
  environment:
    - PORT=8000
    - DATABASE_URL=postgresql://talkdai:talkdai@db:5432/talkdai
    - OPENAI_API_KEY=sk-your-openai-api-key
    - STATIC_FILE_LOCATION=/app/static
    - DIALOG_DATA_PATH=./data/your.csv
    - PROJECT_CONFIG=./data/your.toml

```

## healthcheck(web) + build + networks

```

networks:
  net:
    driver: bridge

services:
  server:
    image: server
    build:
      context: .
      dockerfile: Dockerfile
    volumes:
      # Be aware that indexed data are located in "/chroma/chroma/"
      # Default configuration for persist_directory in chromadb/config.py
      # Read more about deployments: https://docs.trychroma.com/deployment

```

```
- chroma-data:/chroma/chroma

command: "--workers 1 --host 0.0.0.0 --port 8000 --proxy-headers --log-config chromadb/log_config.yml --
timeout-keep-alive 30"

environment:
  - IS_PERSISTENT=TRUE
  - CHROMA_SERVER_AUTHN_PROVIDER=${CHROMA_SERVER_AUTHN_PROVIDER}
  - CHROMA_SERVER_AUTHN_CREDENTIALS_FILE=${CHROMA_SERVER_AUTHN_CREDENTIALS_FILE}
  - CHROMA_SERVER_AUTHN_CREDENTIALS=${CHROMA_SERVER_AUTHN_CREDENTIALS}
  - CHROMA_AUTH_TOKEN_TRANSPORT_HEADER=${CHROMA_AUTH_TOKEN_TRANSPORT_HEADER}
  - PERSIST_DIRECTORY=${PERSIST_DIRECTORY:-/chroma/chroma}
  - CHROMA_OTEL_EXPORTER_ENDPOINT=${CHROMA_OTEL_EXPORTER_ENDPOINT}
  - CHROMA_OTEL_EXPORTER_HEADERS=${CHROMA_OTEL_EXPORTER_HEADERS}
  - CHROMA_OTEL_SERVICE_NAME=${CHROMA_OTEL_SERVICE_NAME}
  - CHROMA_OTEL_GRANULARITY=${CHROMA_OTEL_GRANULARITY}
  - CHROMA_SERVER_NOFILE=${CHROMA_SERVER_NOFILE}

restart: unless-stopped # possible values are: "no", "always", "on-failure", "unless-stopped"

ports:
  - "8000:8000"

healthcheck:
  # Adjust below to match your container port
  test: [ "CMD", "curl", "-f", "http://localhost:8000/api/v1/heartbeat" ]
  interval: 30s
  timeout: 10s
  retries: 3

networks:
  - net

volumes:
  chroma-data:
    driver: local
```