Install n8n on Docker Compose

```
networks:
  ai1:

x-n8n: &service-n8n
  image: n8nio/n8n:latest
  networks: ['ai1']
  environment:
    - DB_TYPE=postgresdb
    - DB_POSTGRESDB_HOST=face
    - DB_POSTGRESDB_USER=${POSTGRES_USER} #use .env file for user
    - DB_POSTGRESDB_PASSWORD=${POSTGRES_PASSWORD} #use .env file for password
    - N8N_DIAGNOSTICS_ENABLED=false
    - N8N_PERSONALIZATION_ENABLED=false
    - N8N_ENCRYPTION_KEY
    - N8N_USER_MANAGEMENT_JWT_SECRET
    - N8N_SECURE_COOKIE=false
  links:
    - postgresdb

x-ollama: &service-ollama
  image: ollama/ollama:latest
  container_name: ollama
  networks: ['ai1']
  restart: always
  ports:
    - 11434:11434
  volumes:
    - /your/local/directory/:/root/.ollama
```

```
x-init-ollama: &init-ollama
  image: ollama/ollama:latest
  networks: ['ai1']
  container_name: ollama-pull-llama
  volumes:
    - /your/local/directory/:/root/.ollama
  entrypoint: /bin/sh
  command:
    - "-c"
    - "sleep 3; OLLAMA_HOST=ollama:11434 ollama pull llama3.1; OLLAMA_HOST=ollama:11434 ollama pull
nomic-embed-text"

services:
  postgresdb:
    image: postgres:16-alpine
    networks: ['ai1']
    restart: always
    ports:
      - 5432:5432
    environment:
      - POSTGRES_USER
      - POSTGRES_PASSWORD
      - POSTGRES_DB
    volumes:
      - /your/local/directory/:/var/lib/postgresql/data
    healthcheck:

      test: ['CMD-SHELL', 'pg_isready -h localhost -U ${POSTGRES_USER} -d ${POSTGRES_DB}']
      interval: 5s
      timeout: 5s
      retries: 10

  n8n-import:
    <<: *service-n8n
    container_name: n8n-import
    entrypoint: /bin/sh
    command:
      - "-c"
      - "n8n import:credentials --separate --input=/backup/credentials && n8n import:workflow --separate --
input=/backup/workflows"
```

```
  volumes:
    - /your/local/directory/:/backup
  depends_on:
    postgresdb:
      condition: service_healthy

n8n:
  <<: *service-n8n
  container_name: n8n
  restart: always
  ports:
    - 5678:5678
  volumes:
    - /your/local/directory/:/home/node/.n8n
    - /your/local/directory/:/backup
    - /your/local/directory/:/data/shared
  depends_on:
    postgresdb:
      condition: service_healthy
    n8n-import:
      condition: service_completed_successfully

qdrant:
  image: qdrant/qdrant
  container_name: qdrant
  networks: ['ai1']
  restart: always
  ports:
    - 6333:6333
  volumes:
    - /your/local/directory/:/qdrant/storage

ollama-cpu:
  profiles: ["cpu"]
  <<: *service-ollama

ollama-gpu:
  profiles: ["gpu-nvidia"]
  <<: *service-ollama
  deploy:
```

```
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: 1
              capabilities: [gpu]

  ollama-pull-llama-cpu:
    profiles: ["cpu"]
    <<: *init-ollama
    depends_on:
      - ollama-cpu

  ollama-pull-llama-gpu:
    profiles: ["gpu-nvidia"]
    <<: *init-ollama
    depends_on:
      - ollama-gpu
```

To run it correctly, you will need to choose to run it with CPU or a GPU

```
docker-compose --profile cpu up -d
```

or

```
docker-compose --profile gpu-nvidia up -d
```

https://github.com/coleam00/ai-agents-masterclass/blob/main/local-ai-packaged/README.md

---

Revision #6
Created 20 November 2024 05:06:57 by Danicus
Updated 20 November 2024 05:21:21 by Danicus