Docker_환경설정_가이드

Docker 환경 설정 가이드

프로젝트: EduMentor AI
목적: Docker를 활용한 데이터베이스 및 전체 시스템 관리
작성일: 2025-10-28

📋 목차

🏗️ Docker Compose 구조

전체 시스템 구성

```
services:
  1. postgres      - PostgreSQL 15 데이터베이스
  2. chromadb      - ChromaDB 벡터 데이터베이스
  3. python-server - FastAPI AI 엔진 (Port 8000)
  4. spring-backend - Spring Boot Backend (Port 8080)
```

서비스별 역할

서비스	포트	역할	의존성
postgres	5432	메타데이터 저장 (User, Material, QA, Problem)	-
chromadb	8001	벡터 임베딩 저장 및 검색	-
python-server	8000	QA + 문제 생성 AI 엔진	chromadb, postgres
spring-backend	8080	REST API 및 비즈니스 로직	postgres, python-server

---

🗄️ 데이터베이스 관리

1. PostgreSQL 설정

docker-compose.yml 설정

```yaml
postgres:
  image: postgres:15
  container_name: edumentor-postgres
  environment:
    POSTGRES_DB: edumentor
    POSTGRES_USER: admin
    POSTGRES_PASSWORD: password
  ports:
    - "5432:5432"
  volumes:
    - postgres_data:/var/lib/postgresql/data
    - ./docker/postgres/init.sql:/docker-entrypoint-initdb.d/init.sql
  healthcheck:
    test: ["CMD-SHELL", "pg_isready -U admin -d edumentor"]
    interval: 10s
```

```
    timeout: 5s
    retries: 5
```

초기화 스크립트 (init.sql)

데이터베이스는 최초 실행 시 자동으로 초기화됩니다:

```
-- 위치: docker/postgres/init.sql

-- 테이블 생성
- users (사용자)
- learning_materials (학습 자료)
- qa_sessions (QA 세션)
- practice_problems (실습 문제)

-- 인덱스 생성
- JSONB GIN 인덱스 (sources, hints, test_cases)
- 성능 최적화 인덱스

-- 샘플 데이터
- instructor1 / password123
- student1 / password123
```

2. ChromaDB 설정

```
chromadb:
  image: chromadb/chroma:latest
  container_name: edumentor-chroma
  ports:
    - "8001:8000"
  volumes:
    - chroma_data:/chroma/data
  environment:
    - IS_PERSISTENT=TRUE
    - ANONYMIZED_TELEMETRY=FALSE
  healthcheck:
    test: ["CMD", "curl", "-f", "http://localhost:8000/api/v1/heartbeat"]
    interval: 10s
    timeout: 5s
    retries: 5
```

3. 데이터 영속성 (Volumes)

```
volumes:
  postgres_data:
    name: edumentor_postgres_data
  chroma_data:
    name: edumentor_chroma_data
```

중요: 데이터는 Docker 볼륨에 저장되어 컨테이너 재시작 후에도 유지됩니다.

🛠️ 개발 환경 설정

1. 환경 변수 설정

```
# 프로젝트 루트에서
cd python-server
cp .env.example .env

# .env 파일 편집
UPSTAGE_API_KEY=your_actual_upstage_api_key
POSTGRES_PASSWORD=your_secure_password  # 운영 환경에서는 반드시 변경!
JWT_SECRET=your_secure_jwt_secret
```

2. 데이터베이스만 실행 (로컬 개발)

```
# PostgreSQL + ChromaDB만 시작
docker-compose up -d postgres chromadb

# 상태 확인
docker-compose ps

# 로그 확인
docker-compose logs -f postgres
docker-compose logs -f chromadb
```

3. Python 서버 로컬 실행

```
# 데이터베이스는 Docker, Python은 로컬
cd python-server
python -m venv venv
source venv/bin/activate  # Windows: venv\Scripts\activate
pip install -r requirements.txt
python main.py
```

접속: http://localhost:8000/docs (FastAPI Swagger UI)

4. Spring Boot 로컬 실행

```
# 데이터베이스는 Docker, Spring Boot는 로컬
cd spring-backend
./gradlew bootRun
```

접속: http://localhost:8080

🚀 운영 환경 배포

1. 전체 시스템 Docker 실행

```
# 전체 빌드 및 실행
docker-compose up --build

# 백그라운드 실행
docker-compose up -d

# 특정 서비스만 재시작
docker-compose restart python-server

# 로그 확인 (실시간)
docker-compose logs -f

# 특정 서비스 로그만
docker-compose logs -f spring-backend
```

2. 시스템 상태 확인

```
# 실행 중인 컨테이너 확인
docker-compose ps

# 헬스체크 상태
docker inspect edumentor-postgres | grep -A 5 Health
docker inspect edumentor-chroma | grep -A 5 Health

# 리소스 사용량
docker stats
```

3. 데이터베이스 접속

PostgreSQL 접속

```bash
# Docker 컨테이너 내부 접속
docker exec -it edumentor-postgres psql -U admin -d edumentor

# 로컬에서 psql 사용 (PostgreSQL 클라이언트 설치 필요)
psql -h localhost -p 5432 -U admin -d edumentor
```

유용한 SQL 명령어

```bash
-- 테이블 목록
\dt

-- 특정 테이블 구조 확인
\d users
\d learning_materials

-- 데이터 조회
SELECT * FROM users;
SELECT * FROM qa_sessions ORDER BY created_at DESC LIMIT 10;

-- 통계 확인
SELECT difficulty, COUNT(*) FROM practice_problems GROUP BY difficulty;
```

4. 데이터 백업 및 복원

백업

```bash
# 전체 데이터베이스 백업
docker exec edumentor-postgres pg_dump -U admin edumentor > backup_$(date +%Y%m%d_%H%M%S).sql

# 특정 테이블만 백업
docker exec edumentor-postgres pg_dump -U admin -t qa_sessions edumentor > qa_backup.sql

# ChromaDB 데이터 백업
docker cp edumentor-chroma:/chroma/data ./chroma_backup
```

복원

```bash
# PostgreSQL 복원
docker exec -i edumentor-postgres psql -U admin -d edumentor < backup.sql

# ChromaDB 복원
docker cp ./chroma_backup edumentor-chroma:/chroma/data
docker-compose restart chromadb
```

🔧 트러블슈팅

문제 1: PostgreSQL 연결 실패

증상:

```
FATAL: password authentication failed for user "admin"
```

해결:

```bash
# 1. 환경 변수 확인
docker-compose config | grep POSTGRES

# 2. 컨테이너 재시작
docker-compose down
docker-compose up -d postgres

# 3. 볼륨 초기화 (데이터 삭제 주의!)
```

```
docker-compose down -v
docker-compose up -d postgres
```

문제 2: ChromaDB 연결 실패

증상:

```
Failed to connect to ChromaDB at localhost:8001
```

해결:

```
# 1. 포트 확인
docker-compose ps chromadb
netstat -an | grep 8001

# 2. 헬스체크 상태
docker inspect edumentor-chroma | grep -A 10 Health

# 3. 로그 확인
docker-compose logs chromadb

# 4. 재시작
docker-compose restart chromadb
```

문제 3: 데이터베이스 초기화 실패

증상:

```
init.sql 스크립트가 실행되지 않음
```

해결:

```
# 1. 볼륨 완전 삭제 후 재생성
docker-compose down -v
docker volume rm edumentor_postgres_data
docker-compose up -d postgres

# 2. 수동 초기화
docker cp docker/postgres/init.sql edumentor-postgres:/tmp/init.sql
docker exec -it edumentor-postgres psql -U admin -d edumentor -f /tmp/init.sql
```

문제 4: Port already in use

증상:

```
Error starting userland proxy: listen tcp4 0.0.0.0:5432: bind: address already in use
```

해결:

```
# 1. 포트 사용 중인 프로세스 확인
lsof -i :5432  # macOS/Linux
netstat -ano | findstr :5432  # Windows

# 2. 프로세스 종료 또는 포트 변경
# docker-compose.yml에서 포트 변경:
# ports:
#   - "5433:5432"  # 로컬 5433 포트로 변경

# 3. 기존 PostgreSQL 중지
sudo systemctl stop postgresql  # Linux
brew services stop postgresql  # macOS
```

문제 5: 디스크 공간 부족

증상:

```
no space left on device
```

해결:

```bash
# 1. Docker 정리
docker system prune -a --volumes

# 2. 사용하지 않는 볼륨 삭제
docker volume ls
docker volume rm <volume_name>

# 3. 사용량 확인
docker system df
```

문제 6: 네트워크 연결 문제

증상:

```
spring-backend cannot connect to postgres
python-server cannot connect to chromadb
```

해결:

```bash
# 1. 네트워크 확인
docker network ls
docker network inspect edumentor-network

# 2. 네트워크 재생성
docker-compose down
docker network rm edumentor-network
docker-compose up -d

# 3. 서비스 간 연결 테스트
docker exec edumentor-backend ping postgres
docker exec python-server curl http://chromadb:8000/api/v1/heartbeat
```

📊 모니터링 및 로그

실시간 로그 확인

```bash
# 전체 로그
docker-compose logs -f

# 특정 서비스만
docker-compose logs -f postgres chromadb

# 최근 100줄만
docker-compose logs --tail=100 python-server

# 타임스탬프 포함
docker-compose logs -f -t
```

리소스 모니터링

```bash
# 실시간 리소스 사용량
docker stats

# 특정 컨테이너만
docker stats edumentor-postgres edumentor-chroma
```

헬스체크 모니터링

```bash
# 모든 컨테이너 헬스체크
docker ps --format "table {{.Names}}\t{{.Status}}"

# 헬스체크 로그
docker inspect --format='{{json .State.Health}}' edumentor-postgres | jq
```

🔐 보안 권장사항

1. 환경 변수 보안

```bash
# .env 파일을 절대 Git에 커밋하지 마세요!
echo ".env" >> .gitignore

# 운영 환경에서는 강력한 비밀번호 사용
POSTGRES_PASSWORD=$(openssl rand -base64 32)
JWT_SECRET=$(openssl rand -base64 64)
```

2. 네트워크 격리

```yaml
# 운영 환경에서는 내부 네트워크 사용
networks:
  frontend:
    driver: bridge
  backend:
    driver: bridge
    internal: true  # 외부 접근 차단
```

3. PostgreSQL 보안

```bash
# SSL 연결 강제
SPRING_DATASOURCE_URL=jdbc:postgresql://postgres:5432/edumentor?ssl=true&sslmode=require

# 비밀번호 정책 강화
# docker/postgres/init.sql에 추가:
ALTER SYSTEM SET password_encryption = 'scram-sha-256';
```

📋 체크리스트

초기 설정

- [ ] Docker 및 Docker Compose 설치 확인
- [ ] .env 파일 생성 및 API 키 설정
- [ ] PostgreSQL 초기화 스크립트 확인
- [ ] 포트 충돌 확인 (5432, 8000, 8001, 8080)

개발 환경

- [ ] `docker-compose up -d postgres chromadb` 실행
- [ ] PostgreSQL 연결 테스트
- [ ] ChromaDB 연결 테스트
- [ ] Python 서버 로컬 실행 확인
- [ ] Spring Boot 로컬 실행 확인

운영 환경

- [ ] 환경 변수 보안 검토
- [ ] 데이터 백업 전략 수립
- [ ] 헬스체크 정상 동작 확인
- [ ] 네트워크 보안 설정

- [ ] 모니터링 시스템 구축

🔗 관련 문서

- README.md - 프로젝트 개요
- 전체시스템아키텍처.md - 시스템 아키텍처
- SpringBoot연동가이드.md - Spring Boot 설정
- Python통합구현가이드.md - Python 서버 설정

작성일: 2025-10-28
최종 수정: 2025-10-28