Dify + Ollama 로컬 스택 구축 런북 (Docker)

| 작성: 2026-09-07
| 대상: 이 PC(JUNOWIN11)에서 대학입시 RAG용 Dify + Ollama 스택을 구축·운영하는 경우
| 관련 문서: Dify 대학입시 로컬 RAG

이 문서는 대학입시 상담 RAG 시스템을 위해 Ollama + Docker Desktop + Dify를 이 PC에 설치·구동한 실제 작업 기록입니다. 다음에 같은 환경을 재현하거나 스택을 다시 올릴 때 그대로 따라 할 수 있도록 명령과 함정을 함께 남깁니다.

구축 결과 요약

구성요소	버전	접속 / 경로
Ollama	0.33.3	http://localhost:11434
Ollama 모델	qwen2.5(상담 LLM), bge-m3(임베딩)	—
Docker Desktop	4.89.0 / Engine 29.7.2	C:₩Program Files₩Docker₩Docker
Dify	최신(shallow clone)	http://localhost:8088
Dify 소스	—	C:₩Developments₩dify

컨테이너 15개(api, web, worker, worker_beat, postgres, redis, weaviate, nginx, sandbox, plugin_daemon 등)가 함께 뜹니다.

1. Ollama 설치

```
winget install --id Ollama.Ollama --accept-source-agreements --accept-package-agreements --silent
```

설치하면 백그라운드 서비스가 자동 실행되어 http://localhost:11434가 열립니다. 새 터미널에서 모델을 받습니다.

```
ollama pull qwen2.5      # 상담용 LLM (한국어)
ollama pull bge-m3       # 임베딩 (RAG 검색)
ollama list              # 확인
```

2. Docker Desktop 설치

```
winget install --id Docker.DockerDesktop --accept-source-agreements --accept-package-agreements --silent
```

• 이 PC는 WSL2가 이미 설치(2.6.3, Ubuntu, 기본 버전 2)되어 있어 재부팅 없이 바로 사용 가능했습니다. WSL2가 없는 환경에서는 최초 설치 시 WSL2 활성화 + 재부팅이 1회 필요할 수 있습니다.
• CLI(docker)는 기본 PATH에 없을 수 있으므로, 스크립트에서는 다음을 앞에 붙입니다.

```
$env:Path += ";C:₩Program Files₩Docker₩Docker₩resources₩bin"
```

엔진 기동:

```
Start-Process "C:₩Program Files₩Docker₩Docker₩Docker Desktop.exe"
# 엔진이 뜰 때까지 대기 후 확인
docker version --format '{{.Server.Version}}'
```

3. Dify 내려받기 및 실행

```
git clone --depth 1 https://github.com/langgenius/dify.git C:₩Developments₩dify
Copy-Item C:₩Developments₩dify₩docker₩.env.example C:₩Developments₩dify₩docker₩.env
cd C:₩Developments₩dify₩docker
docker compose up -d
```

4. 포트 충돌 회피 (중요)

Dify nginx는 기본으로 호스트 80/443을 사용합니다. 그런데 이 PC는 Caddy가 이미 80/443을 점유(dev.tictechtoeai.com 등 서빙)하고 있어, 그대로 두면 요청이 Caddy로 가서 Server: Caddy 응답과 308 → https 리다이렉트가 나옵니다.

진단 방법:

```
# 응답 헤더에 Server 확인
curl.exe -s -D - -o NUL http://127.0.0.1/
# 호스트 80/443 점유 프로세스 확인
Get-NetTCPConnection -LocalPort 80,443 -State Listen | Select LocalPort,OwningProcess
```

해결: C:₩Developments₩dify₩docker₩.env에서 노출 포트를 비어 있는 값으로 변경합니다.

```
EXPOSE_NGINX_PORT=8088
EXPOSE_NGINX_SSL_PORT=8449
```

변경 후 nginx만 재생성:

```
cd C:₩Developments₩dify₩docker
docker compose up -d nginx
docker port docker-nginx-1      # 80/tcp -> 0.0.0.0:8088 확인
```

이제 http://localhost:8088 로 접속합니다. (재클론·재빌드 시 이 포트 설정을 다시 넣어야 합니다.)

5. 초기 설정 및 Ollama 연결

1. 브라우저에서 http://localhost:8088/install → 관리자 계정(이메일/비밀번호) 생성
2. 설정 → 모델 공급자 → Ollama 플러그인 설치
3. 모델 추가:
   - LLM: 이름 qwen2.5, Base URL http://host.docker.internal:11434
   - Text Embedding: 이름 bge-m3, Base URL http://host.docker.internal:11434
4. 시스템 모델 설정에서 기본 Reasoning qwen2.5, Embedding bge-m3 지정

컨테이너에서 호스트의 Ollama에 접근할 때는 반드시 http://host.docker.internal:11434를 씁니다. localhost를 넣으면 컨테이너 자기 자신을 가리켜 연결이 실패합니다. 연결은 다음으로 미리 검증할 수 있습니다.

```
docker exec docker-api-1 sh -c "curl -s http://host.docker.internal:11434/api/version"
# {"version":"0.33.3"} 가 나오면 정상
```

6. 상태 확인 · 운영 명령

```
$env:Path += ";C:₩Program Files₩Docker₩Docker₩resources₩bin"
cd C:₩Developments₩dify₩docker

docker compose ps                  # 컨테이너 상태
curl.exe -s http://localhost:8088/console/api/setup
#   {"step":"not_started"}  → 아직 관리자 계정 없음
```

```
#   {"step":"finished"}     → 계정 생성 완료

docker compose logs -f api        # api 로그
docker compose logs -f worker     # 임베딩·비동기 작업 로그
```

7. 시작 / 정지 / 완전 종료

```
cd C:₩Developments₩dify₩docker

docker compose up -d      # 시작 (설정·데이터 볼륨 유지)
docker compose stop       # 컨테이너만 정지 (데이터 보존)
docker compose down       # 컨테이너 제거 (볼륨은 유지, -v 붙이면 데이터까지 삭제 — 주의)
```

Docker Desktop 자체 종료:

```
docker desktop stop
```

관리자 계정, Ollama 공급자 설정 등은 모두 Docker 볼륨(postgres 등)에 영구 저장되므로, 컨테이너를 정지하거나 PC를 꺼도 사라지지 않습니다. 다시 docker compose up -d 하면 그대로 복구됩니다.

자주 걸리는 부분

• Server: Caddy / 308 https 리다이렉트: 호스트 80/443을 Caddy가 점유. 위 4장대로 포트를 8088/8449로 변경.
• 모델 저장 시 Connection error: Base URL을 host.docker.internal로 넣었는지, 모델 이름이 ollama list 결과와 정확히 일치하는지 확인.
• /console/api/setup 502: api 컨테이너가 아직 기동 중. docker inspect --format '{{.State.Health.Status}}' docker-api-1가 healthy가 될 때까지 대기.
• 문서 임베딩이 멈춤: 임베딩은 worker가 처리하므로 docker compose logs -f worker 확인.

다음 단계

입시 자료(모집요강 PDF)가 준비되면 지식베이스 생성 → 임베딩(bge-m3, High Quality) → 검색 테스트 → 상담 앱(Chatflow) 연결 순서로 진행합니다. 자세한 설계는 Dify 대학입시 로컬 RAG 문서를 참고하세요.