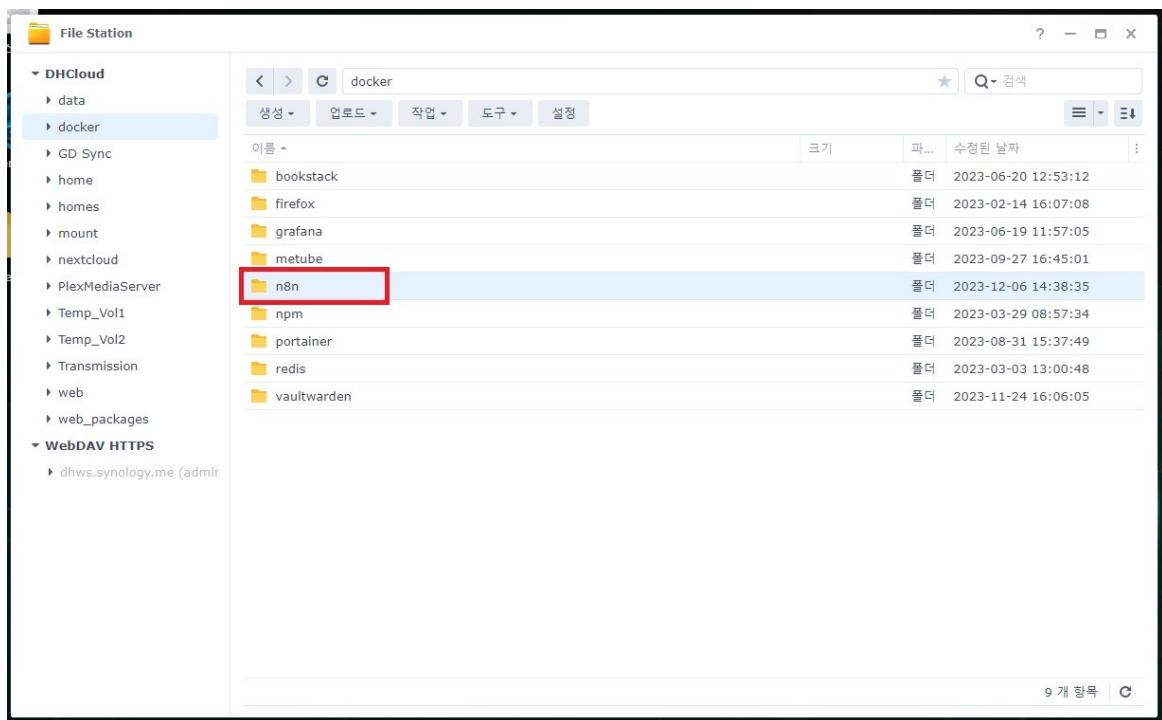
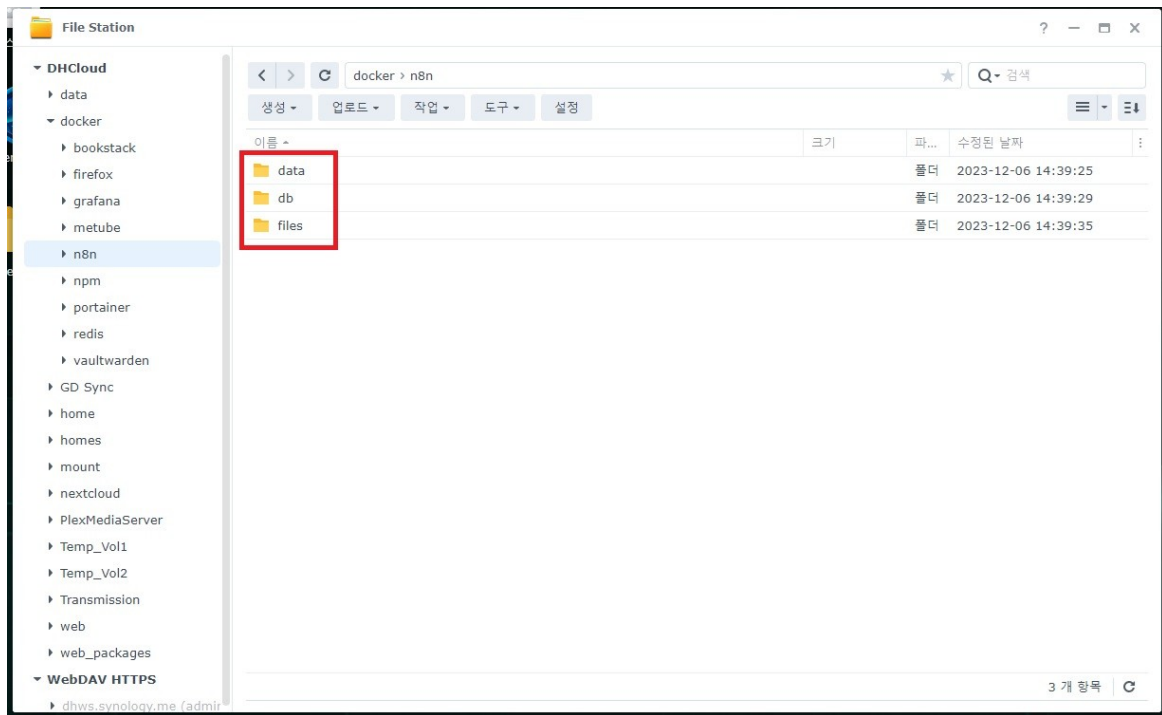


설치

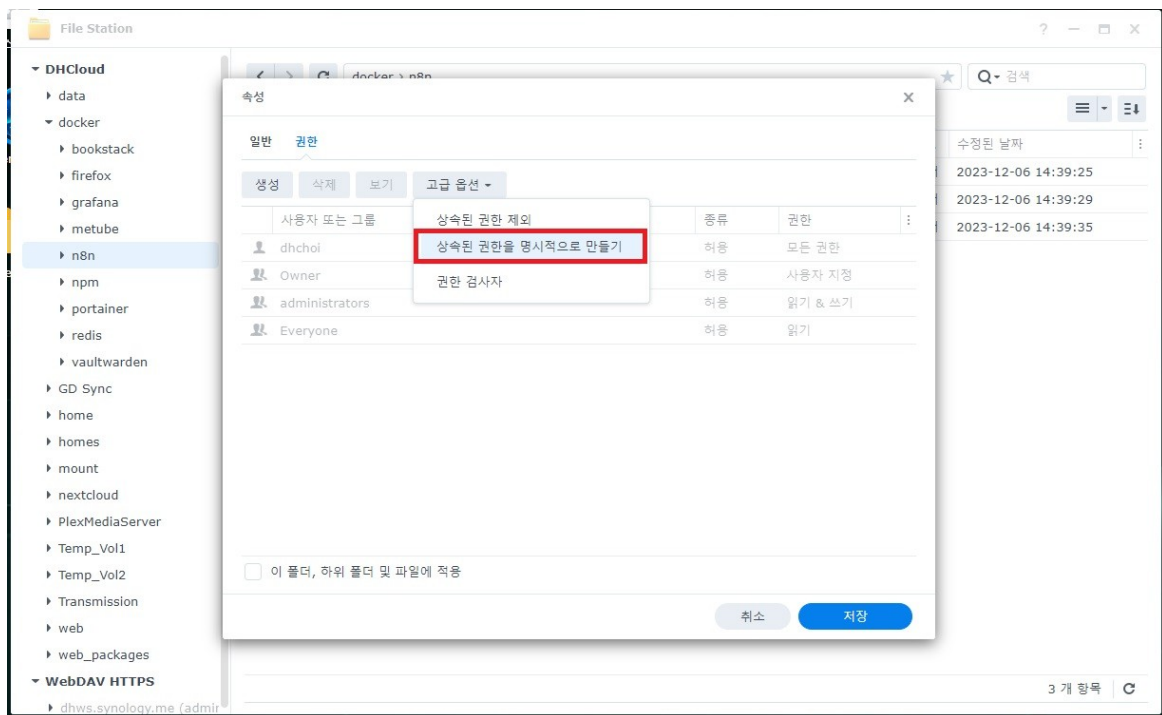
시놀로지에 n8n + Postgres DB Docker를 Portainer를 활용한 설치 방법을 설명합니다. Portainer를 사용하지 않을 시 yaml 파일 생성해 설치하시면 됩니다.

1. Nginx Proxy Manager 같은 역방향 프록시를 설정합니다.
2. 볼륨을 Mapping할 "n8n"과 하위에 "data", "db", "files" 3개의 디렉토리를 생성합니다. 제 경우 "/volume1/docker/"하위에 생성을 하였습니다. (redis 사용 시 "n8n" 밑에 "redis" 디렉토리를 하나 더 추가해 줍니다.)





3. 생성한 "n8n" 속성 (오른쪽 마우스 클릭)에 들어가 "권한" 탭으로 이동 후 "고급옵션"에서 "상속된 권한을 명시적으로 만들기"를 선택하고 "저장"을 누릅니다.



4. 목록에서 "Everyone"의 "편집"을 선택하고, "읽기", "쓰기"의 모든 옵션을 체크 합니다.

권한 편집기

사용자 또는 그룹: Everyone

상속 대상: <없음>

종류: 허용

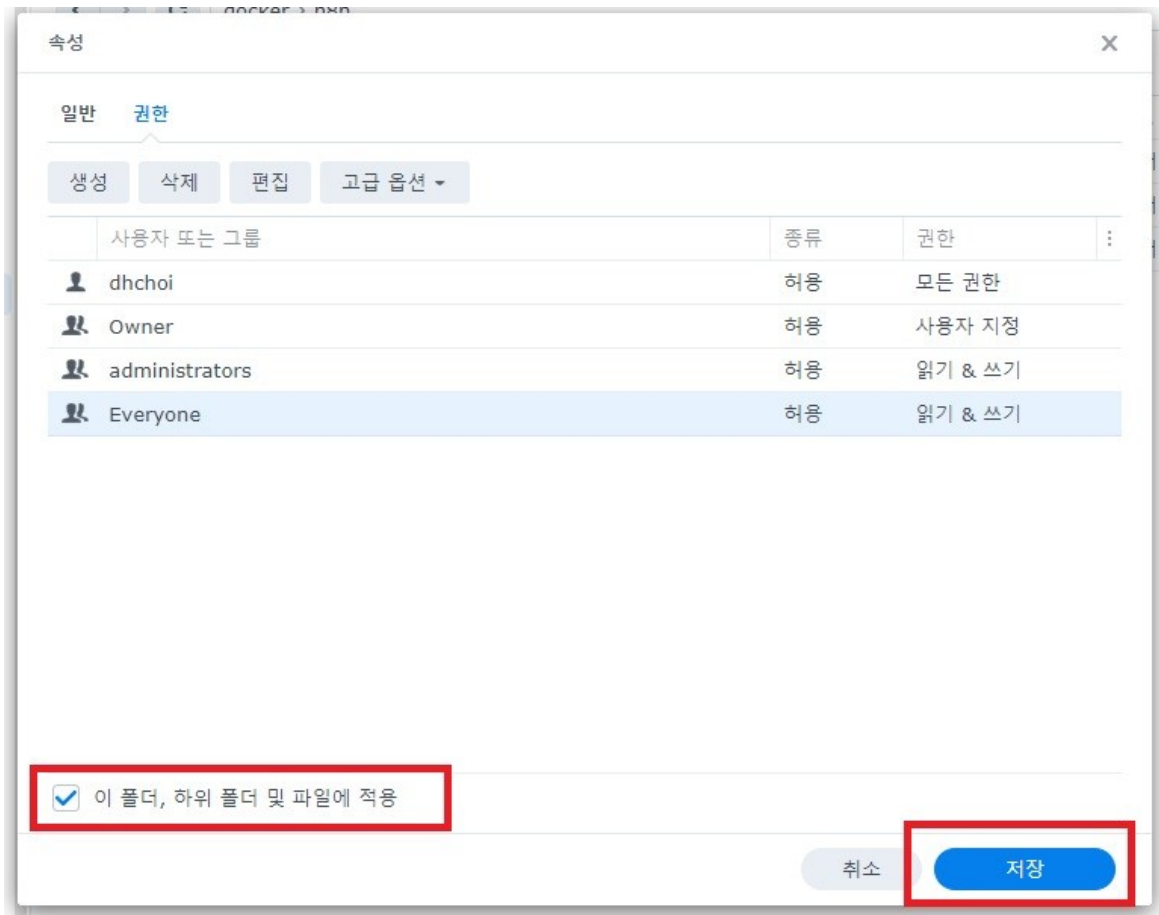
적용 대상: 모두

권한

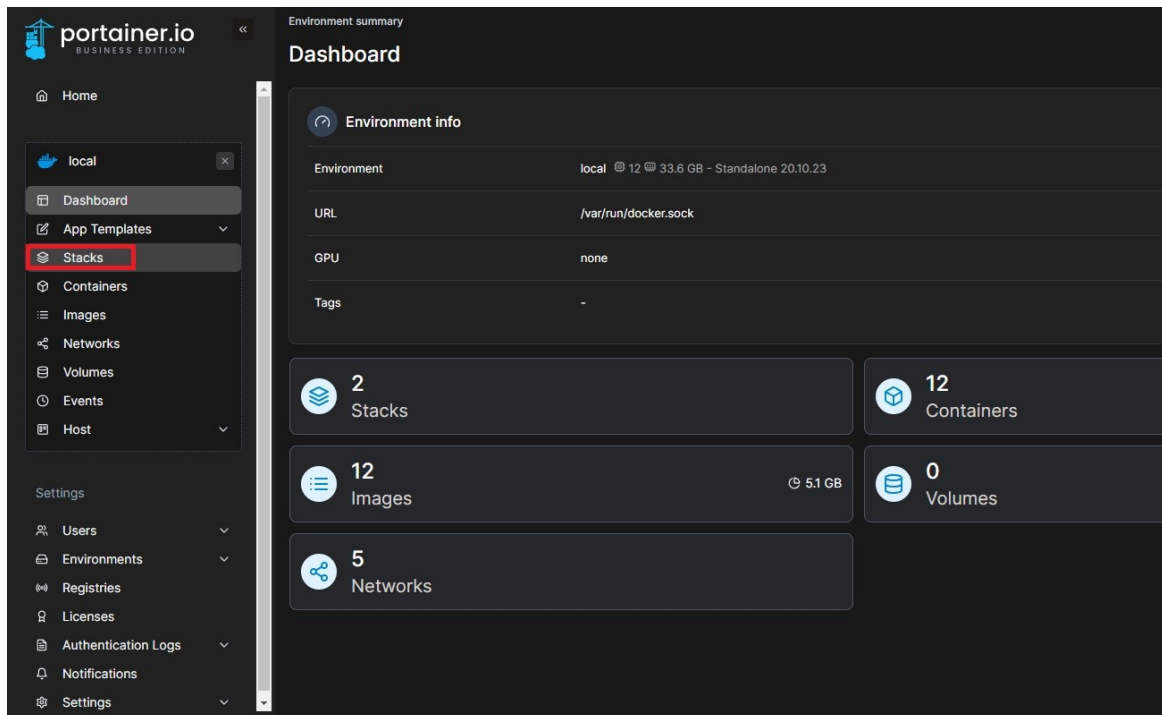
- ☐ 관리
 - ☐ 권한 변경
 - ☐ 소유권 가져오기
- ☒ 읽기
 - ☒ 폴더 탐색/파일 실행
 - ☒ 폴더 나열/데이터 읽기
 - ☒ 읽기 속성
 - ☒ 확장된 읽기 속성
 - ☒ 읽기 권한
- ☒ 쓰기

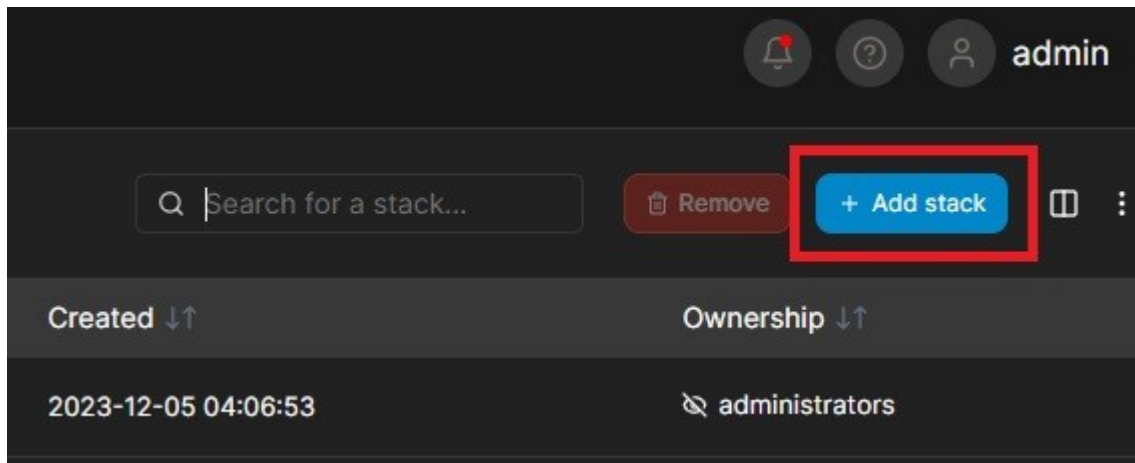
취소 완료

5. "이 폴더, 하위 폴더 및 파일에 적용"을 선택하고 "저장"을 누릅니다.

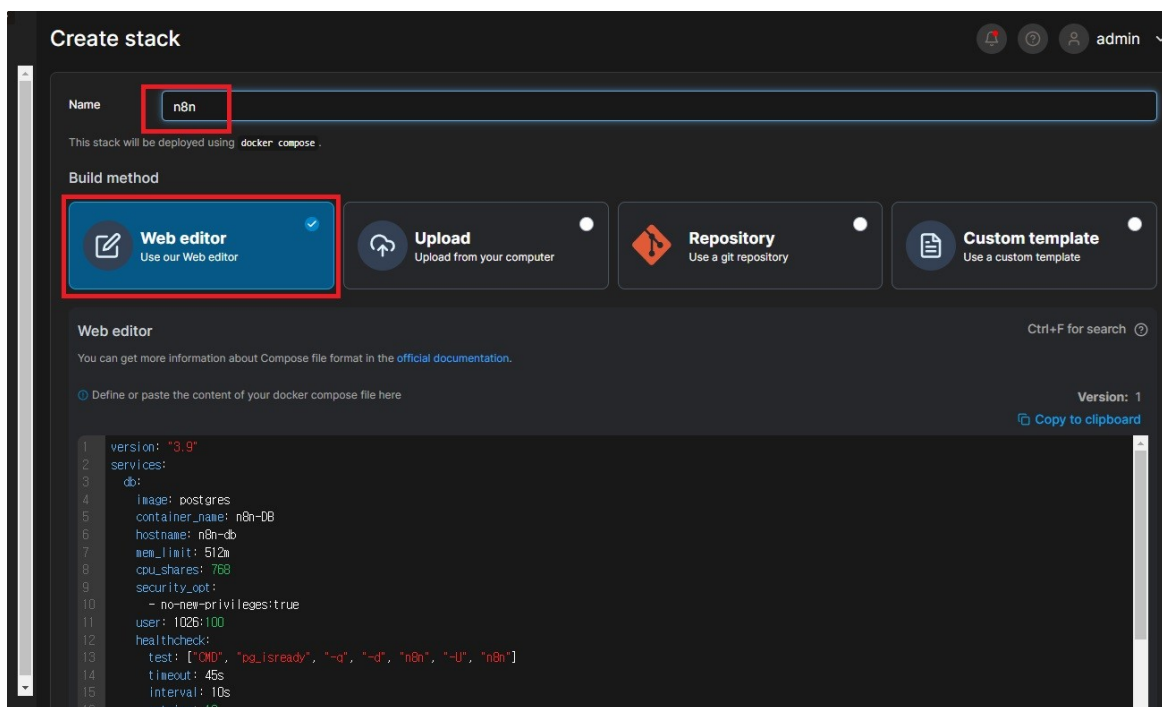


6. Portainer의 "Stacks"로 이동하여, "Add stack"을 선택 합니다.





7. "Name"에 원하시는 컨테이너 이름을 입력하고, "Web editor"를 선택해 yaml 문장을 작성 합니다.



```
version: "3.9"
services:
  db:
    image: postgres
    container_name: n8n-DB
    hostname: n8n-db
    mem_limit: 512m
    cpu_shares: 768
    security_opt:
      - no-new-privileges:true
    user: 1026:100
    healthcheck:
      test: ["CMD", "pg_isready", "-q", "-d", "n8n", "-U", "n8n"]
      timeout: 45s
      interval: 10s
      retries: 10
```

```

volumes:
  - /volume1/docker/n8n/db:/var/lib/postgresql/data:rw
environment:
  TZ: Asia/Seoul
  POSTGRES_DB: n8n
  POSTGRES_USER: n8n
  POSTGRES_PASSWORD: DB_PASSWORD //원하시는 DB패스워드를 입력합니다.
restart: on-failure:5

n8n:
  image: n8nio/n8n:latest
  container_name: n8n
  hostname: n8n
  mem_limit: 1g
  cpu_shares: 768
  security_opt:
    - no-new-privileges:true
  ports:
    - 5678:5678
  volumes:
    - /volume1/docker/n8n/data:/home/node/.n8n:rw
    - /volume1/docker/n8n/files:/files:rw
  environment:
    N8N_HOST: n8n_도메인주소 //사용하실 n8n 도메인을 입력하고, http는 생략합니다.
    N8N_PORT: 5678
    N8N_PROTOCOL: https
    NODE_ENV: production
    WEBHOOK_URL: https://n8n_도메인주소 //사용하실 n8n 도메인을 입력합니다.
    GENERIC_TIMEZONE: Asia/Seoul
    TZ: Asia/Seoul
    DB_TYPE: postgresdb
    DB_POSTGRESDB_DATABASE: n8n
    DB_POSTGRESDB_HOST: n8n-db
    DB_POSTGRESDB_PORT: 5432
    DB_POSTGRESDB_USER: n8n
    DB_POSTGRESDB_PASSWORD: DB_PASSWORD //위에서 설정한 DB패스워드를 입력합니다.
  restart: on-failure:5
  depends_on:
    db:
      condition: service_healthy

```

redis 사용을 원할 시 아래 yaml을 사용합니다.

```

version: "3.9"
services:
  db:
    image: postgres
    container_name: n8n-DB
    hostname: n8n-db
    mem_limit: 512m

```

```
cpu_shares: 768
security_opt:
  - no-new-privileges:true
user: 1026:100
healthcheck:
  test: ["CMD", "pg_isready", "-q", "-d", "n8n", "-U", "n8n"]
  timeout: 45s
  interval: 10s
  retries: 10
volumes:
  - /volume1/docker/n8n/db:/var/lib/postgresql/data:rw
environment:
  TZ: Asia/Seoul
  POSTGRES_DB: n8n
  POSTGRES_USER: n8n
  POSTGRES_PASSWORD: DB_PASSWORD //원하시는 DB패스워드를 입력합니다.
restart: on-failure:5
```

redis Option Start

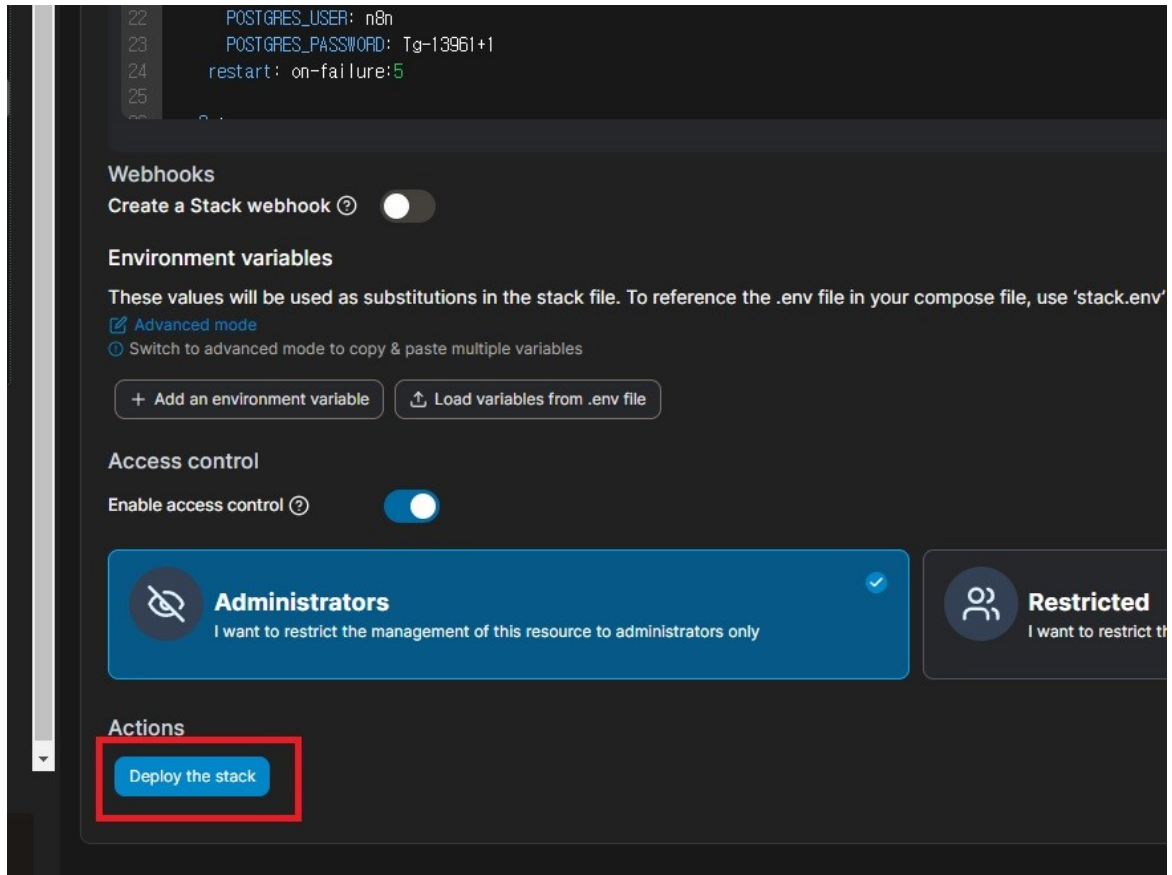
```
redis:
  image: redis
  container_name: n8n-redis
  restart: always
  volumes:
    - /volume1/docker/n8n/redis:/data:rw
  healthcheck:
    test: ['CMD', 'redis-cli', 'ping']
    interval: 5s
    timeout: 5s
    retries: 10
```

redis Option End

```
n8n:
  image: n8nio/n8n:latest
  container_name: n8n
  hostname: n8n
  mem_limit: 1g
  cpu_shares: 768
  security_opt:
    - no-new-privileges:true
  ports:
    - 5678:5678
  volumes:
    - /volume1/docker/n8n/data:/home/node/.n8n:rw
    - /volume1/docker/n8n/files:/files:rw
  environment:
    N8N_HOST: n8n_도메인주소 //사용하실 n8n 도메인을 입력하고, http는 생략합니다.
    N8N_PORT: 5678
    N8N_PROTOCOL: https
    NODE_ENV: production
```

```
WEBHOOK_URL: https://n8n_도메인주소 //사용하실 n8n 도메인을 입력합니다.
# redis Option Start
EXECUTIONS_MODE: queue
EXECUTIONS_QUEUE: redis
QUEUE_HEALTH_CHECK_ACTIVE: true
QUEUE_BULL_REDIS_HOST: n8n-redis
QUEUE_BULL_REDIS_PORT: 6379
# redis Option End
GENERIC_TIMEZONE: Asia/Seoul
TZ: Asia/Seoul
DB_TYPE: postgresdb
DB_POSTGRESDB_DATABASE: n8n
DB_POSTGRESDB_HOST: n8n-db
DB_POSTGRESDB_PORT: 5432
DB_POSTGRESDB_USER: n8n
DB_POSTGRESDB_PASSWORD: DB_PASSWORD //위에서 설정한 DB패스워드를 입력합니다.
restart: on-failure:5
depends_on:
  db:
    condition: service_healthy
  redis:
    condition: service_healthy
links:
  - n8n-DB
  - n8n-redis # redis Option
```

8. yaml 작성이 완료 됐으면 "Deploy the stack"을 눌러 컨테이너를 생성 합니다.



🔄버전 #11

★생성 6 12월 2023 06:00:17, Admin

✎수정 7 12월 2023 00:54:21, Admin