

♦ Ściaga – podstawowe komendy Docker Compose

Wszystkie komendy wykonujesz w katalogu, gdzie leży `docker-compose.yml`.

```
docker compose up          # uruchom stack, logi w konsoli (foreground)
docker compose up -d       # uruchom w tle (detached)

docker compose ps          # pokaż działające usługi z tego compose
docker compose logs        # logi wszystkich usług
docker compose logs db     # logi tylko usługi "db"
docker compose logs -f db  # logi "db" na żywo (follow)

docker compose exec db bash # wejdź do działającego kontenera usługi "db" z
powłoką bash
docker compose exec db sh   # alternatywnie sh, jeśli nie ma bash

docker compose down        # zatrzymaj i usuń kontenery + sieć
docker compose down -v     # to samo, ale dodatkowo usuwa wolumeny

docker volume ls           # lista wolumenów Dockera
docker ps -a               # wszystkie kontenery (również zatrzymane)
```

Zadanie 1 – MariaDB / MySQL z docker run

Cel: poznać bazę w kontenerze i zmienne środowiskowe.

1. Uruchom kontener z `mariadb:latest` komendą `docker run`:
 - ustaw `MYSQL_ROOT_PASSWORD=pass`,
 - opcjonalnie zmapuj port `3306:3306` na hosta.
2. Sprawdź logi kontenera (`docker logs NAZWA`), żeby upewnić się, że MariaDB wystartowała.
3. Zatrzymaj kontener (`docker stop NAZWA`).

Kryteria zaliczenia:

- W logach widać, że serwer jest „ready for connections”.
- Rozumiesz, że hasło roota pochodzi ze zmiennej `MYSQL_ROOT_PASSWORD`.

Na koniec zadania zrób:

- Zapisz historię użytych poleceń (np. w `zadanie1-commands.txt` lub w `README`) i wykonaj commit w Gicie.
- Dodaj **screenshots** pokazujące działające zadanie (np. logi z uruchomionej bazy, `docker ps`).
- Dołącz aktualny plik `docker-compose.yml` (może być pusty lub z komentarzem, że nie był jeszcze używany w tym zadaniu).

- Przygotuj krótki opis, jak wykonałeś zadanie, i **zapisz go jako PDF** (np. `zadanie1-raport.pdf`) w repozytorium.
-

Zadanie 2 – Named volume dla MariaDB (bez compose)

Cel: zrozumieć persystencję danych.

1. Utwórz **named volume** np. `db-vol`:
`docker volume create db-vol`
2. Uruchom kontener MariaDB tak, aby:
 - miał ustawione `MYSQL_ROOT_PASSWORD=pass`,
 - volume `db-vol` był podpięty do `/var/lib/mysql`.
3. Wejdź do kontenera (`docker exec -it NAZWA bash` lub `sh`) i utwórz testową bazę, np. `kurs_docker`.
4. Zatrzymaj i **usuń** kontener (wolumenu **nie** usuwaj).
5. Uruchom **nowy** kontener MariaDB, znów z volume `db-vol` pod `/var/lib/mysql`.
6. Sprawdź, czy baza `kurs_docker` dalej istnieje.

Kryteria zaliczenia:

- Po wymianie kontenera baza nadal jest dostępna.
- Rozumiesz, że dane są w wolumenie, nie w kontenerze.

Na koniec zadania zrób:

- Zapisz wszystkie polecenia użyte w tym zadaniu w pliku (np. `zadanie2-commands.txt`) i zacommituj w Gicie.
 - Dodaj **screenshots**: np. lista wolumenów, potwierdzenie istnienia bazy po restarcie kontenera.
 - Dołącz aktualny plik `docker-compose.yml` (może zawierać np. szkic przyszłej konfiguracji z komentarzem).
 - Przygotuj krótki raport z wnioskami o persystencji danych i **zapisz go jako PDF** (np. `zadanie2-raport.pdf`) w repo.
-

Zadanie 3 – Pierwszy docker-compose.yml (tylko MariaDB)

Cel: przenieść konfigurację z `docker run` do Compose.

1. W nowym katalogu projektu utwórz plik `docker-compose.yml`.

2. Zdefiniuj **jedną usługę db**:

- `obraz mariadb:latest`,
- `environment: MYSQL_ROOT_PASSWORD: pass`,
- `volume db-vol:/var/lib/mysql`.

3. Na dole pliku dodaj sekcję:

```
volumes:  
  db-vol:
```

4. Uruchom stack:

```
docker compose up -d
```

5. Sprawdź:

- `docker compose ps` – czy usługa db działa,
- `docker volume ls` – czy istnieje volume odpowiadający `db-vol`.

Kryteria zaliczenia:

- Jednym `docker compose up -d` odpalasz bazę z persystencją.
- Po `docker compose down` i ponownym `up` dane z bazy się zachowują.

Na koniec zadania zrób:

- Zapisz wszystkie polecenia w `zadanie3-commands.txt` (lub innym pliku) i zatwierdź w Gicie.
- Dodaj **screenshots**: np. wynik `docker compose ps`, `docker volume ls`.
- Załącz aktualny plik `docker-compose.yml` (to już główny artefakt tego zadania).
- Przygotuj krótki opis konfiguracji i działania oraz **zapisz go jako PDF** (`zadanie3-raport.pdf`) w repo.

Zadanie 4 – phpMyAdmin jako osobny kontener (bez compose)

Cel: zobaczyć, jak aplikacja łączy się z bazą w innym kontenerze.

1. Upewnij się, że MariaDB z Zadania 2 lub 3 działa.
2. Uruchom kontener `phpmyadmin/phpmyadmin:latest`:
 - `zmapuj port 8082:80`,
 - ustaw zmienne środowiskowe:
 - `PMA_HOST` – host bazy (IP lub nazwa kontenera z MariaDB),

- PMA_USER=root,
- PMA_PASSWORD=pass.

3. Wejdź na `http://localhost:8082`.

4. Zaloguj się danymi root / pass i sprawdź, czy widzisz bazę.

Kryteria zaliczenia:

- phpMyAdmin się uruchamia i jest dostępny przez przeglądarkę.
- Widzisz bazę MariaDB i możesz przeglądać tabele.

Na koniec zadania zrób:

- Zapisz historię poleceń (`zadanie4-commands.txt`) i zrób commit w Gicie.
 - Dodaj **screenshots** z działającego phpMyAdmin (ekran logowania i widok bazy).
 - Dołącz aktualny `docker-compose.yml` (może być jeszcze tylko z usługą db, ale powinien odzwierciedlać stan repo w tym momencie).
 - Przygotuj krótki raport z opisem, jak skonfigurowałeś phpMyAdmin i **zapisz go jako PDF** (`zadanie4-raport.pdf`).
-

Zadanie 5 – `docker-compose.yml` z dwiema usługami

Cel: przenieść konfigurację phpMyAdmin + MariaDB do jednego pliku compose.

1. W katalogu projektu rozbuduj `docker-compose.yml`, dodając usługę `phpmyadmin`.
2. W pliku mają być:

Usługa db:

```
db:
  image: mariadb:latest
  environment:
    MYSQL_ROOT_PASSWORD: pass
  volumes:
    - db-vol:/var/lib/mysql
```

Usługa phpmyadmin:

```
phpmyadmin:
  image: phpmyadmin/phpmyadmin:latest
  ports:
    - "8082:80"
  environment:
    PMA_HOST: db
    PMA_USER: root
    PMA_PASSWORD: pass
```

3. Uruchom:

```
docker compose up -d
```

4. Sprawdź:

- `docker compose ps` – czy działają obie usługi,
- `http://localhost:8082` – czy logowanie działa bez zmian konfiguracji.

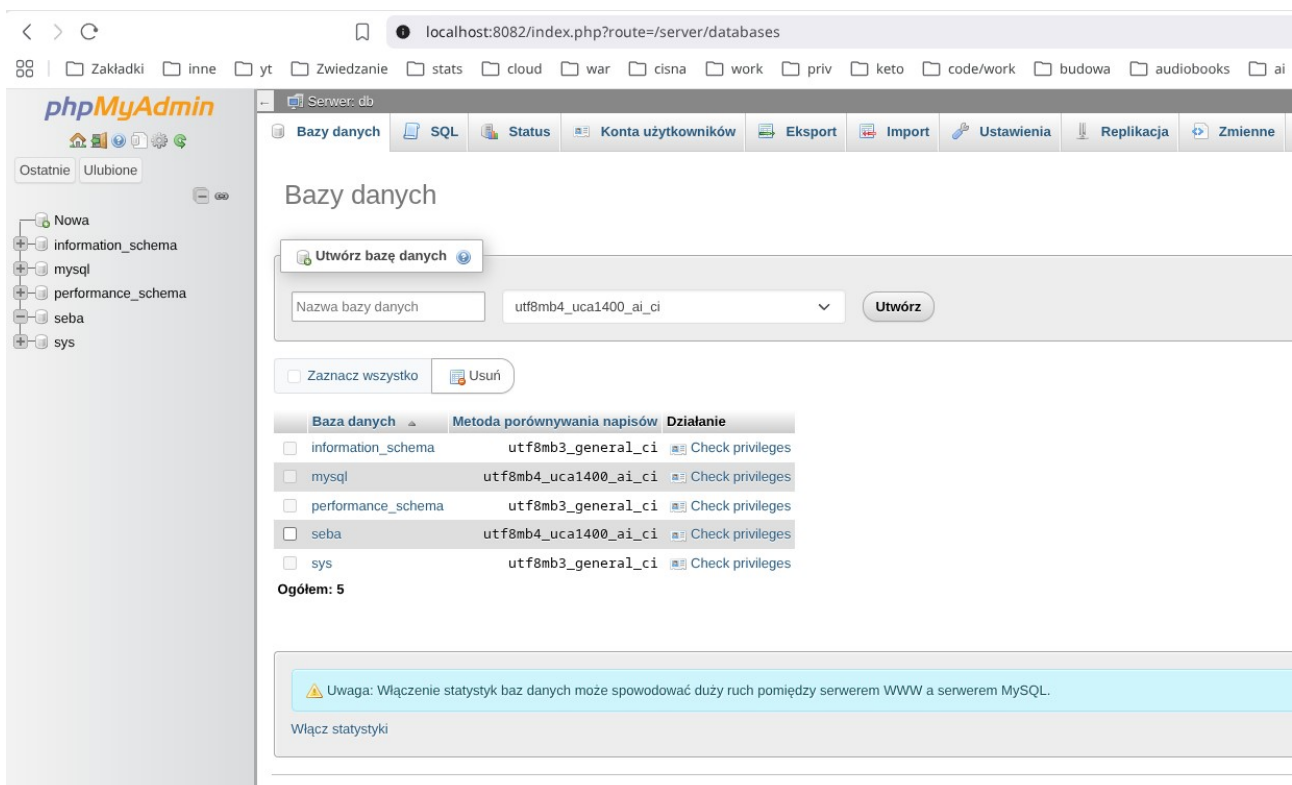
Kryteria zaliczenia:

- Compose startuje **obie** usługi jednym poleceniem.
- phpMyAdmin łączy się z bazą używając nazwy `db` (wewnętrzna sieć compose).

Na koniec zadania zrób:

- Zapisz listę wszystkich użytych poleceń w pliku (np. `zadanie5-commands.txt`) i `zacommituj`.
- Dodaj **screenshots**: `docker compose ps`, widok phpMyAdmin po zalogowaniu.
- Dołącz aktualny plik `docker-compose.yml` z dwiema usługami.
- Przygotuj krótki raport (jak działa sieć między usługami) i **zapisz go jako PDF** (`zadanie5-raport.pdf`) w repo.

Na koniec powinieneś mieć dostęp do PhpMyAdmin z przeglądarki z działającym połączeniem do bazy



W razie problemów mam gotowe rozwiązanie