Homework: Network, Security, and Orchestration of Docker

Description

Sebuah startup ingin membangun aplikasi berbasis web dengan arsitektur berikut:

1. Frontend: Menggunakan Nginx untuk melayani halaman statis.
2. Backend: API sederhana berbasis Python Flask.
3. Database: PostgreSQL untuk menyimpan data aplikasi.
4. Networking: Backend dan database harus saling terhubung dengan jaringan privat.
5. Volume: Database PostgreSQL memerlukan penyimpanan data yang persisten.
6. ~~Scaling: Backend harus bisa di-scale hingga 3 replika.~~

Tugas Peserta

1. Buat Docker Compose file untuk menjalankan semua service.
2. Pastikan Nginx dapat diakses melalui browser.
3. Gunakan Docker Volume untuk menyimpan data PostgreSQL.
4. ~~Gunakan Docker Swarm untuk scaling backend menjadi 3 replika.~~
5. Pastikan semua container saling terhubung melalui jaringan

---

Getting Started

System Design

Rancangan sistem terdiri dari 4 komponen (service) sbb:

- **pg-greeting**

  Database PostgreSQL

  ref: https://hub.docker.com/_/postgres

- **pgadmin-greeting**

Web-based app untuk me-manage atau administrasi database postgres

ref: https://hub.docker.com/r/dpage/pgadmin4

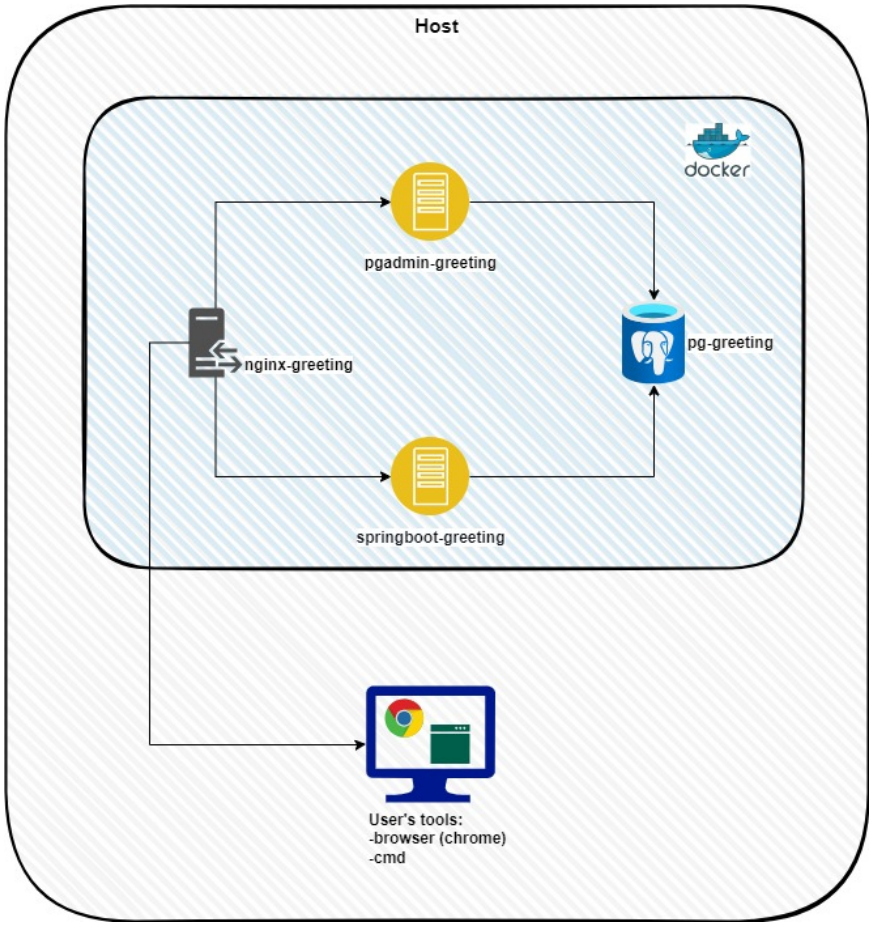
▼ **springboot-greeting**

Aplikasi rest-api berbasis Java-Springboot yang akan menampilkan pesan “hello <nama>” dari database

▼ **nginx-greeting**

Reverse proxy server, sebagai entry-point untuk mengakses sistem. `nginx-greeting` ini satu-satunya service yang akan di-ekspose port-nya ke public/host

Diagram



Dependencies

- Docker
- Browser (chrome, etc)
- cmd, powershell, etc
- Java 17, maven

Installing

- Download source code:

https://github.com/galihi5/devops-digitalskola-greeting

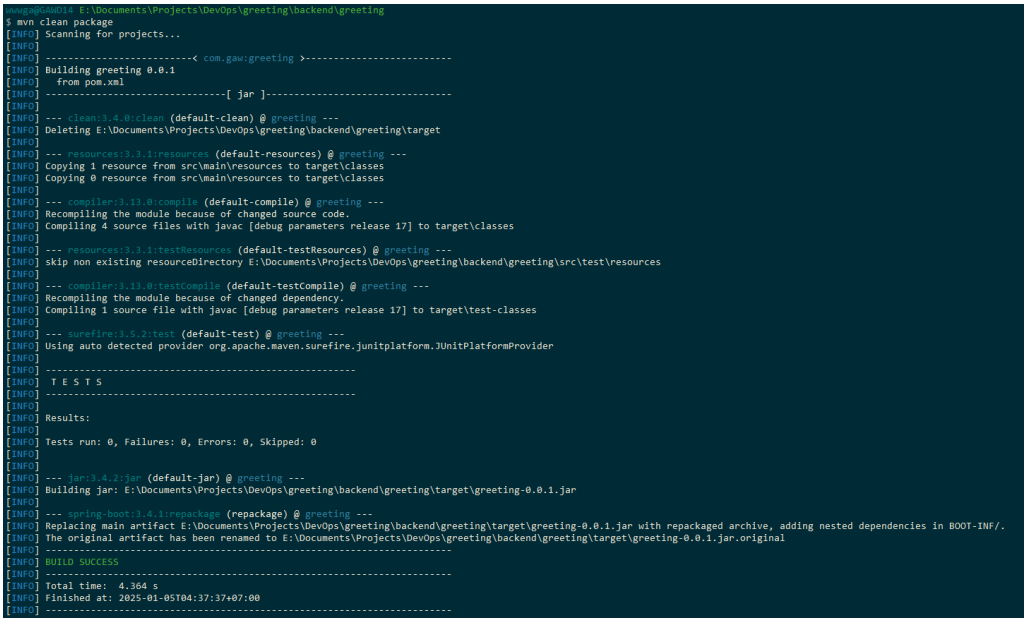
```
greeting
|    .env
|    docker-compose.yml
├──backend
|    |    Dockerfile
|    └──greeting
|        └──target
|                greeting-0.0.1.jar
├──db
├──pgadmin
└──proxy
        default.conf
```

- Add `127.0.0.1 pgadmin.localhost` to file `hosts`

Executing

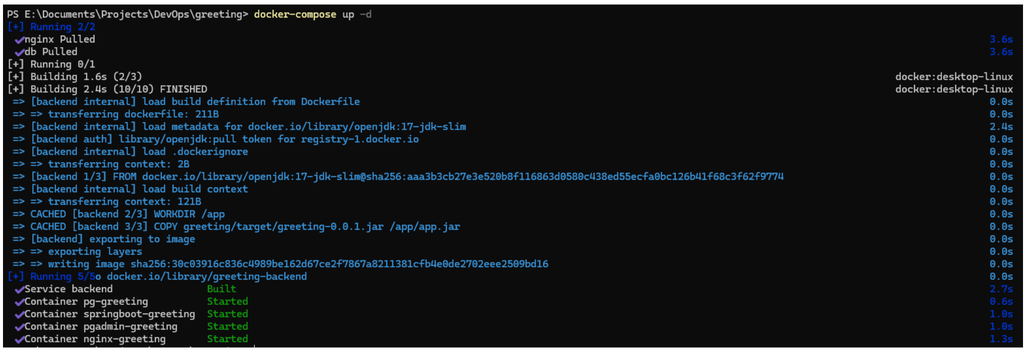
1. build `springboot-greeting`

   ```
   cd backend\greeting
   mvn clean package
   ```



2. Run `docker-compose`

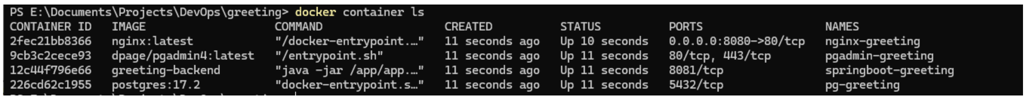
   ```
   docker-compose up -d
   ```



```
PS E:\Documents\Projects\DevOps\greeting> docker-compose up -d
[+] Running 2/2
 ✔nginx Pulled                                                                                      3.6s
 ✔db Pulled                                                                                         3.6s
[+] Running 0/1
[+] Building 1.6s (2/3)                                                              docker:desktop-linux
[+] Building 2.4s (10/10) FINISHED                                                   docker:desktop-linux
 => [backend internal] load build definition from Dockerfile                                        0.0s
 => => transferring dockerfile: 211B                                                                0.0s
 => [backend internal] load metadata for docker.io/library/openjdk:17-jdk-slim                      2.4s
 => [backend auth] library/openjdk:pull token for registry-1.docker.io                              0.0s
 => [backend internal] load .dockerignore                                                           0.0s
 => => transferring context: 2B                                                                     0.0s
 => [backend 1/3] FROM docker.io/library/openjdk:17-jdk-slim@sha256:aaa3b3cb27e3e520b8f116863d0580c438ed55ecfa0bc126b41f68c3f62f9774  0.0s
 => [backend internal] load build context                                                           0.0s
 => => transferring context: 121B                                                                   0.0s
 => CACHED [backend 2/3] WORKDIR /app                                                               0.0s
 => CACHED [backend 3/3] COPY greeting/target/greeting-0.0.1.jar /app/app.jar                       0.0s
 => [backend] exporting to image                                                                    0.0s
 => => exporting layers                                                                             0.0s
 => => writing image sha256:30c03916c836c4989be162d67ce2f7867a8211381cfb4e0de2702eee2509bd16        0.0s
[+] Running 5/5o docker.io/library/greeting-backend                                                 0.0s
 ✔Service backend                 Built                                                             2.7s
 ✔Container pg-greeting           Started                                                           0.6s
 ✔Container springboot-greeting   Started                                                           1.0s
 ✔Container pgadmin-greeting      Started                                                           1.0s
 ✔Container nginx-greeting        Started                                                           1.3s
```

3. Verify that services is running

```
docker container ls
```



```
PS E:\Documents\Projects\DevOps\greeting> docker container ls
CONTAINER ID   IMAGE                   COMMAND                  CREATED          STATUS          PORTS                  NAMES
2fec21bb8366   nginx:latest            "/docker-entrypoint.…"   11 seconds ago   Up 10 seconds   0.0.0.0:8080->80/tcp   nginx-greeting
9cb3c2cece93   dpage/pgadmin4:latest   "/entrypoint.sh"         11 seconds ago   Up 11 seconds   80/tcp, 443/tcp        pgadmin-greeting
12c44f796e66   greeting-backend        "java -jar /app/app.…"   11 seconds ago   Up 11 seconds   8081/tcp               springboot-greeting
226cd62c1955   postgres:17.2           "docker-entrypoint.s…"   11 seconds ago   Up 11 seconds   5432/tcp               pg-greeting
```

Demo

Demo project ini bisa dilihat di

https://youtu.be/IlqZYiP6j38

Execute greeting

- [x] -

Akses pgAdmin melalui Nginx

- [x] ~~Akses http://pgadmin.localhost:8080/ melalui browser~~
- [x] ~~Login sesuai email/password di `.env`~~
- [x] ~~Create connection to `pg-greeting`~~
- [x] ~~Create table `users`~~

```
CREATE TABLE public.users (
    user_id serial4 NOT NULL,
    username varchar(50) NOT NULL,
    created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
    CONSTRAINT users_pkey PRIMARY KEY (user_id),
    CONSTRAINT users_username_key UNIQUE (username)
);
```

- [x] ~~Insert value~~

```
insert into users (username) values ('John');
```

Call service `springboot-greeting` melalui Nginx

- [x] ~~Call http://localhost:8080/hello melalui web browser~~

Restart `docker-compose`

Memastikan bahwa data terakhir masih tersimpan setelah docker di-restart

- ~~`docker-compose down` lalu `docker-compose up -d`~~
- ~~login ke `pgadmin` untuk melihat data "John" masih ada~~
- ~~Call http://localhost:8080/hello melalui web browser~~

Memastikan perubahan data dari `pgAdmin` terlihat pada `springboot-greeting`

- ~~Call http://localhost:8080/hello multiple times (execute bat script dari cmd dengan delay 1 detik)~~

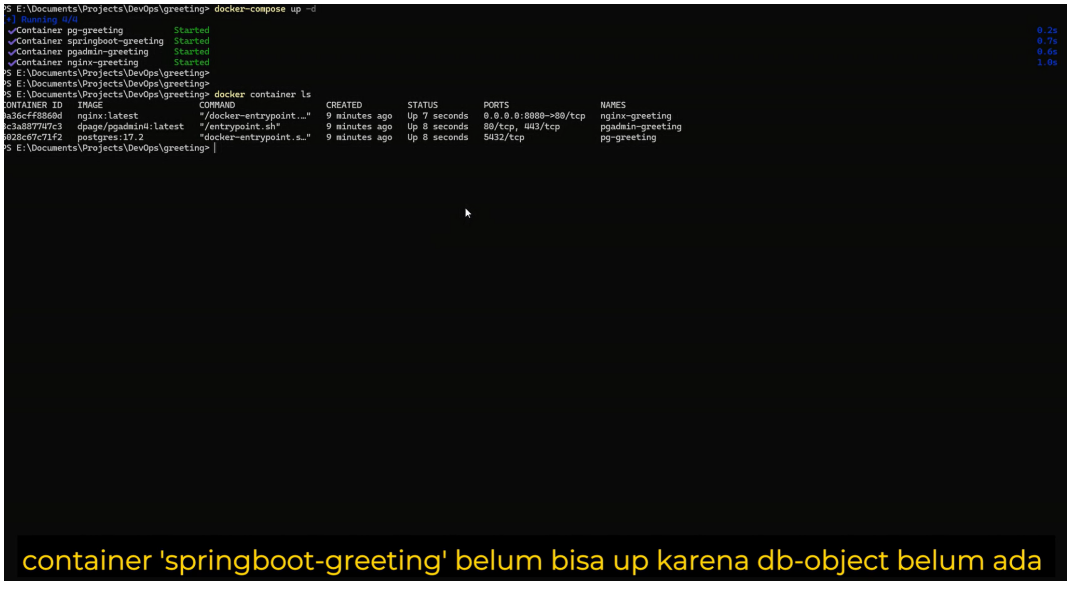
```
@ECHO OFF
ECHO "call http://localhost:8080/hello with delay 1 sec"

for /l %%x in (1,1,100) do (
  curl http://localhost:8080/hello
  timeout /t 1 /nobreak >nul
  ECHO .
)

ECHO "DONE"
```

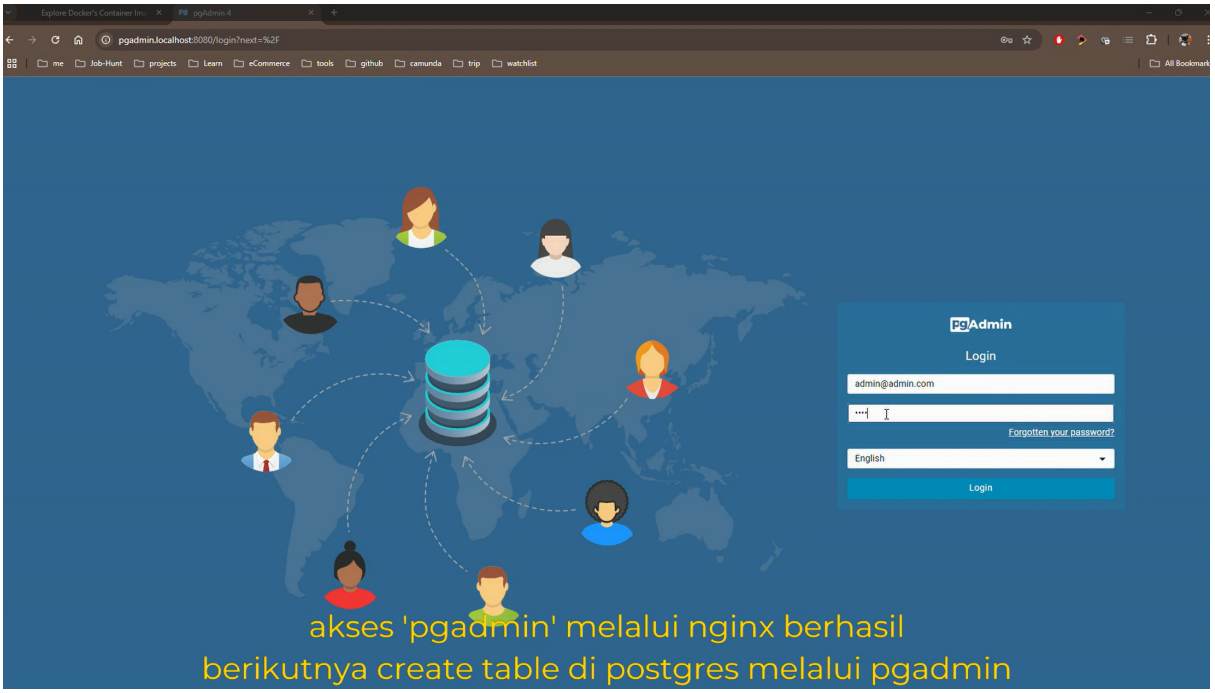
- ~~Insert new value melalui `pgAdmin`~~

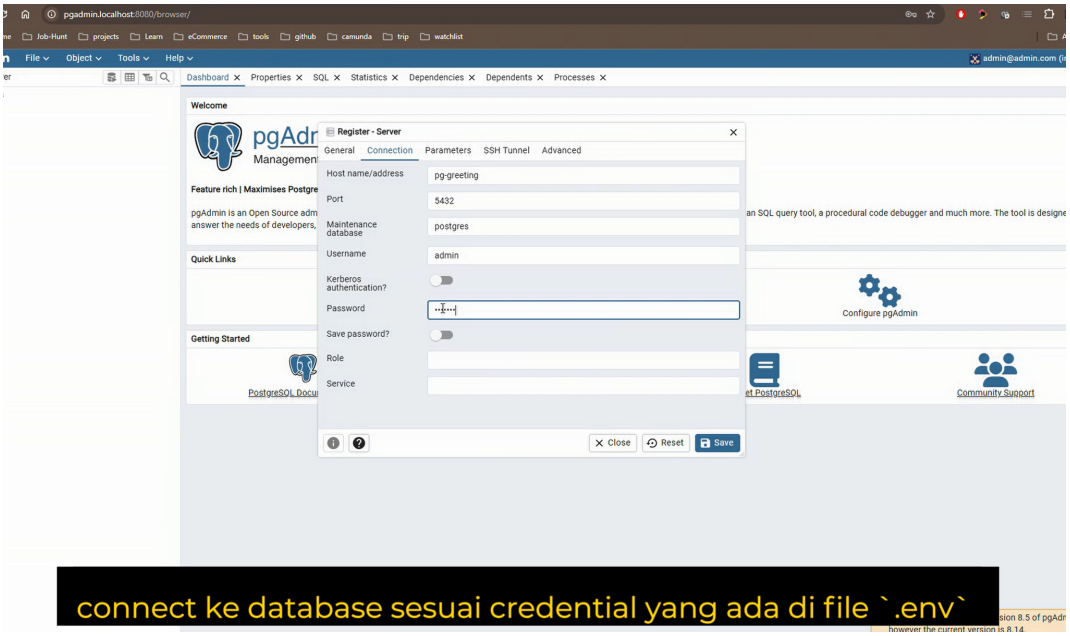
```
insert into users (username) values ('Doe');
```

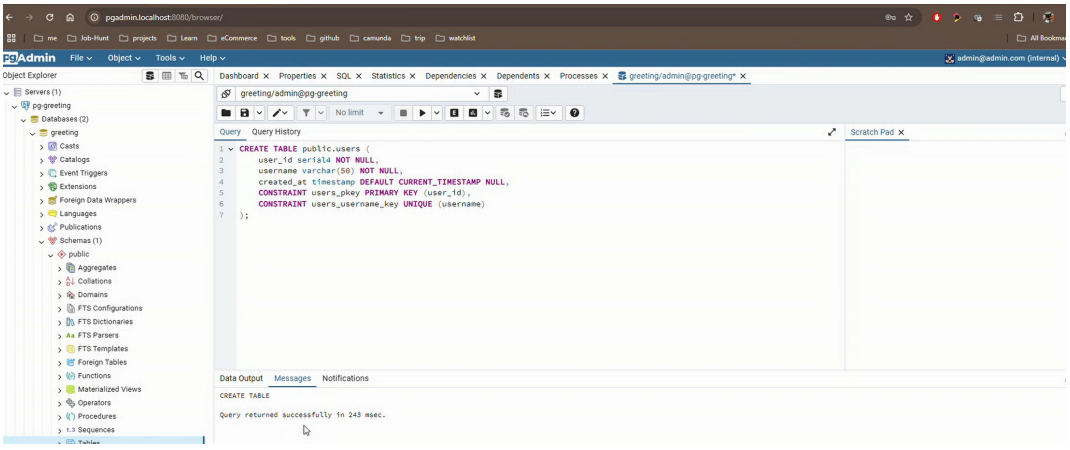
- ~~Perubahan response langsung terlihat pada hasil `curl`~~

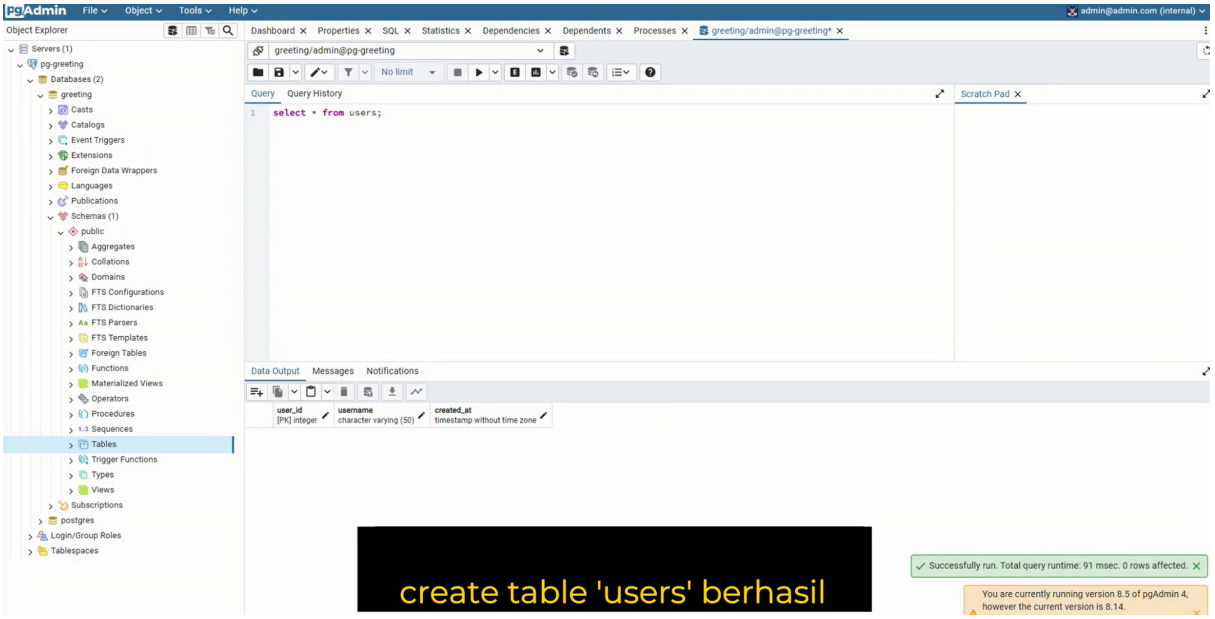
Screenshot

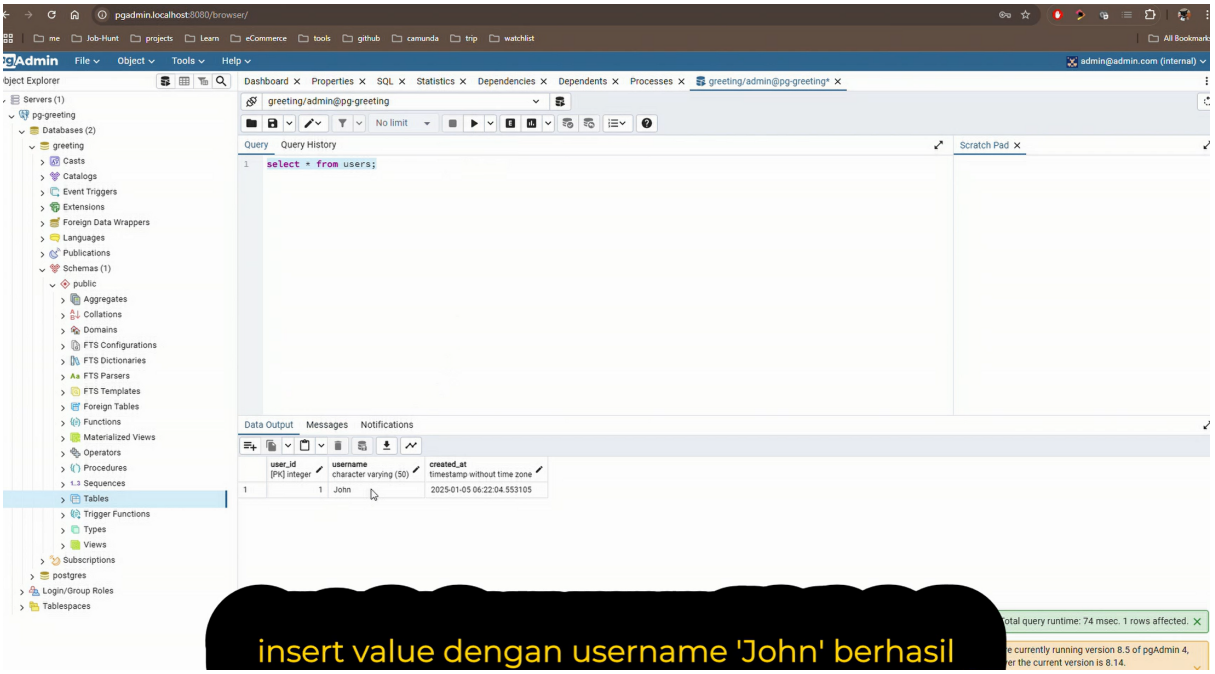


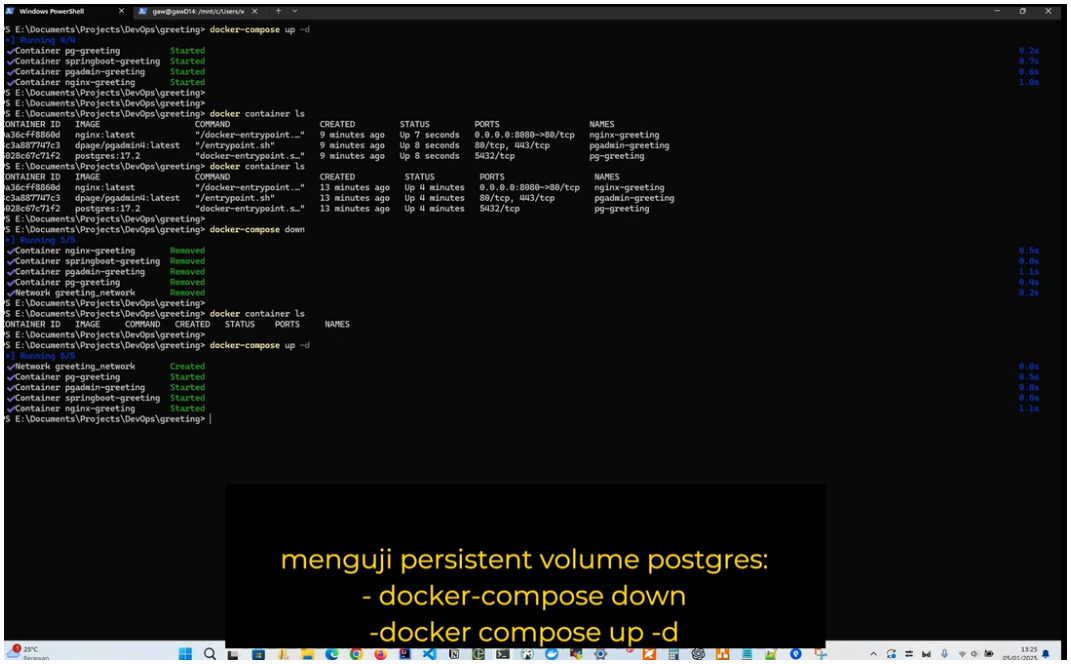

pgAdmin
Login
admin@admin.com
Forgotten your password?
English
Login
akses 'pgadmin' melalui nginx berhasil
berikutnya create table di postgres melalui pgadmin

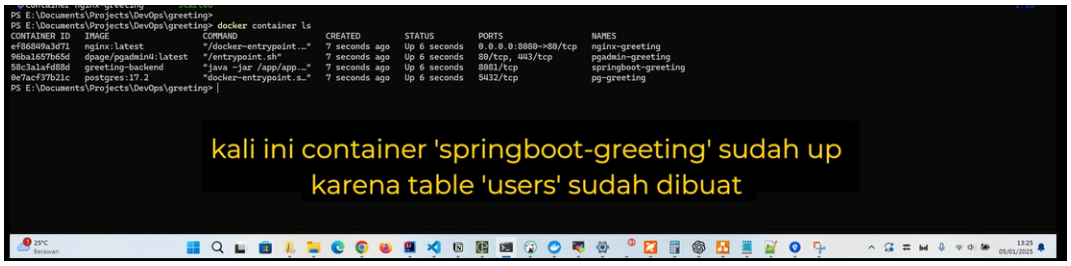

Register - Server
General
Connection
Parameters
SSH Tunnel
Advanced
Host name/address
pg-greeting
Port
5432
Maintenance database
postgres
Username
admin
Password
Save password?
Role
Service
Close
Reset
Save
connect ke database sesuai credential yang ada di file `.env`

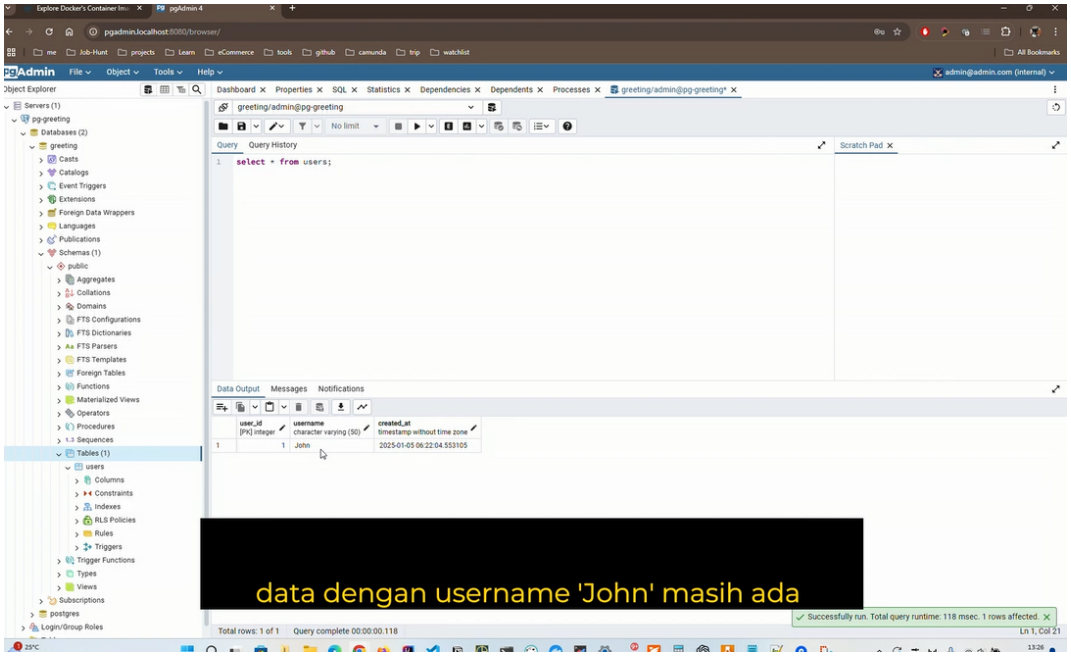

CREATE TABLE public.users (
user_id serial4 NOT NULL,
username varchar(50) NOT NULL,
created_at timestamp DEFAULT CURRENT_TIMESTAMP NULL,
CONSTRAINT users_pkey PRIMARY KEY (user_id),
CONSTRAINT users_username_key UNIQUE (username)
);
Data Output
Messages
Notifications
CREATE TABLE
Query returned successfully in 243 msec.

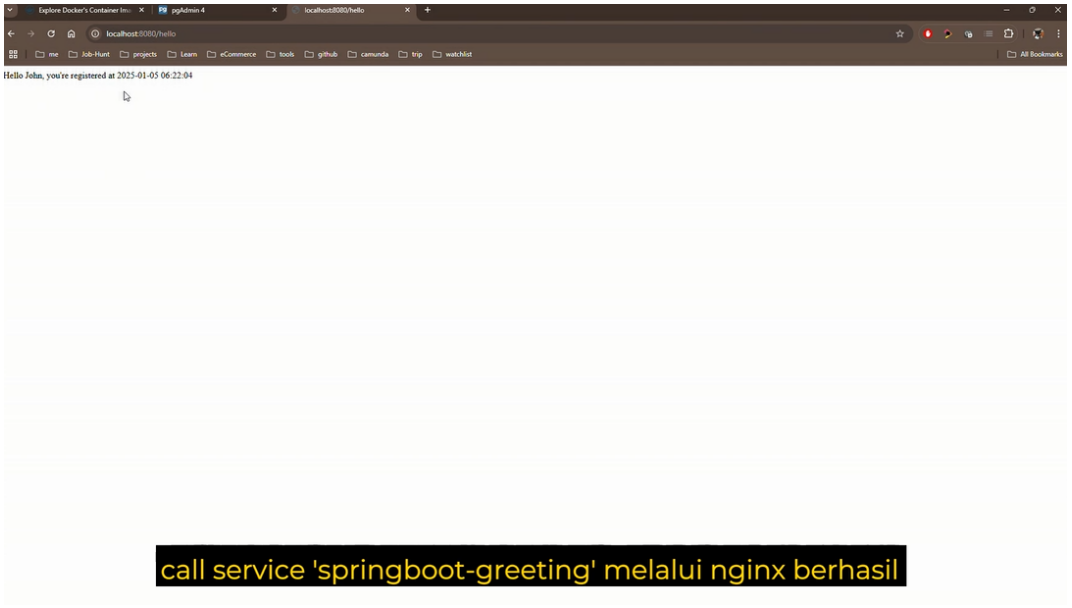

select * from users;
create table 'users' berhasil

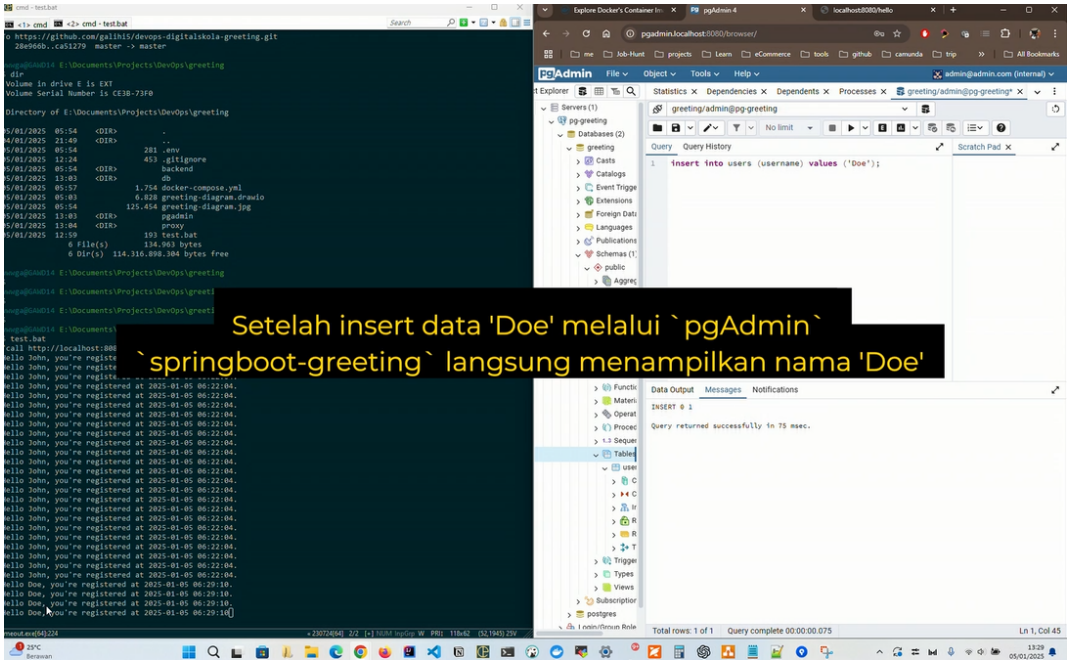

select * from users;
insert value dengan username 'John' berhasil


menguji persistent volume postgres:
- docker-compose down
-docker compose up -d


kali ini container 'springboot-greeting' sudah up
karena table 'users' sudah dibuat


data dengan username 'John' masih ada


Hello John, you're registered at 2025-01-05 06:22:04
call service 'springboot-greeting' melalui nginx berhasil


Setelah insert data 'Doe' melalui `pgAdmin`
`springboot-greeting` langsung menampilkan nama 'Doe'