

Deskripsi:

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. (tidak perlu)

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. (tidak perlu)
5. Pastikan semua container saling terhubung melalui jaringan privat.

Jawaban

Berikut ini berisikan gambar dan hasil dari yang telah saya lakukan. Sebagai catatan saya menambahkan monitoring.

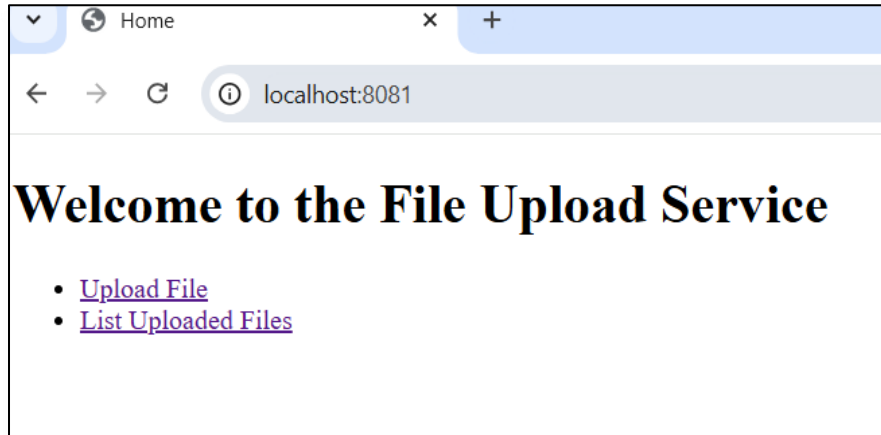
```
linux_2@T3S4PHS8:~/monitoring/server-agent$ tree
.
├── crud
│   ├── backend
│   │   ├── Dockerfile
│   │   ├── app.py
│   │   ├── config.py
│   │   └── requirements.txt
│   ├── docker-compose.yml
│   ├── home
│   │   └── linux_2
│   │       ├── monitoring
│   │       │   ├── server-agent
│   │       │   │   ├── crud
│   │       │   │   │   ├── nginx
│   │       │   │   │   └── nginx.conf
│   │       ├── migrations
│   │       ├── nginx
│   │       │   └── nginx.conf
│   │       └── upload
│   └── docker-compose.yml
└── 13 directories, 7 files
```

Tampilan struktur direktori

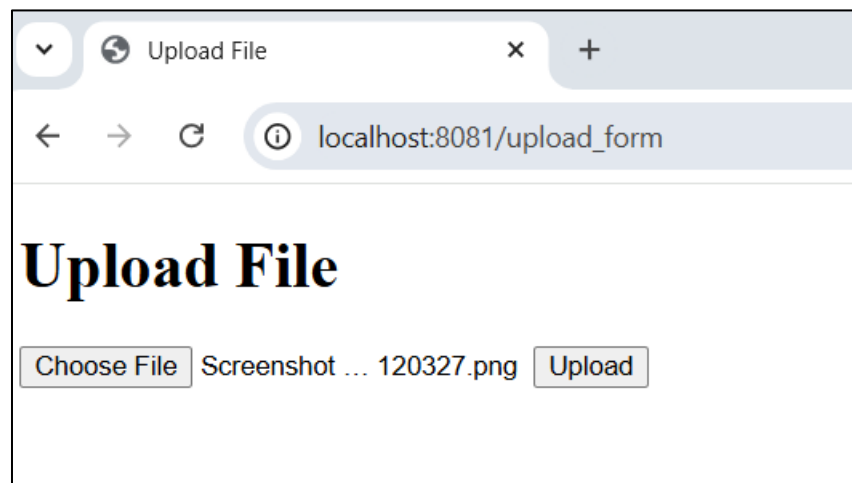
```
linux_2@T3S4PHS8:~/monitoring/server-agent/crud$ docker-compose ps -a
```

Name	Command	State	Ports
backend	python app.py	Up	0.0.0.0:5000->5000/tcp, :::5000->5000/tcp
nginx	/docker-entrypoint.sh nginx ...	Up	0.0.0.0:8081->80/tcp, :::8081->80/tcp
node-exporter	/bin/node_exporter	Up	0.0.0.0:9100->9100/tcp, :::9100->9100/tcp
postgres	docker-entrypoint.sh postgres	Up	0.0.0.0:5432->5432/tcp, :::5432->5432/tcp

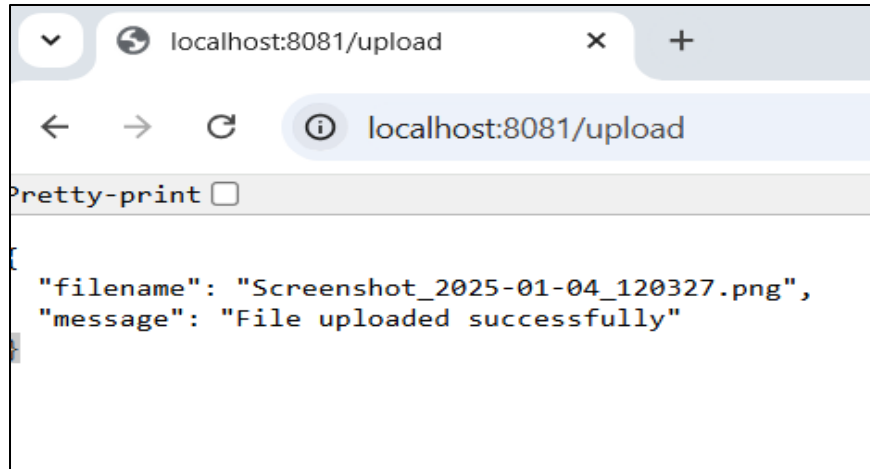
Tampilan list container



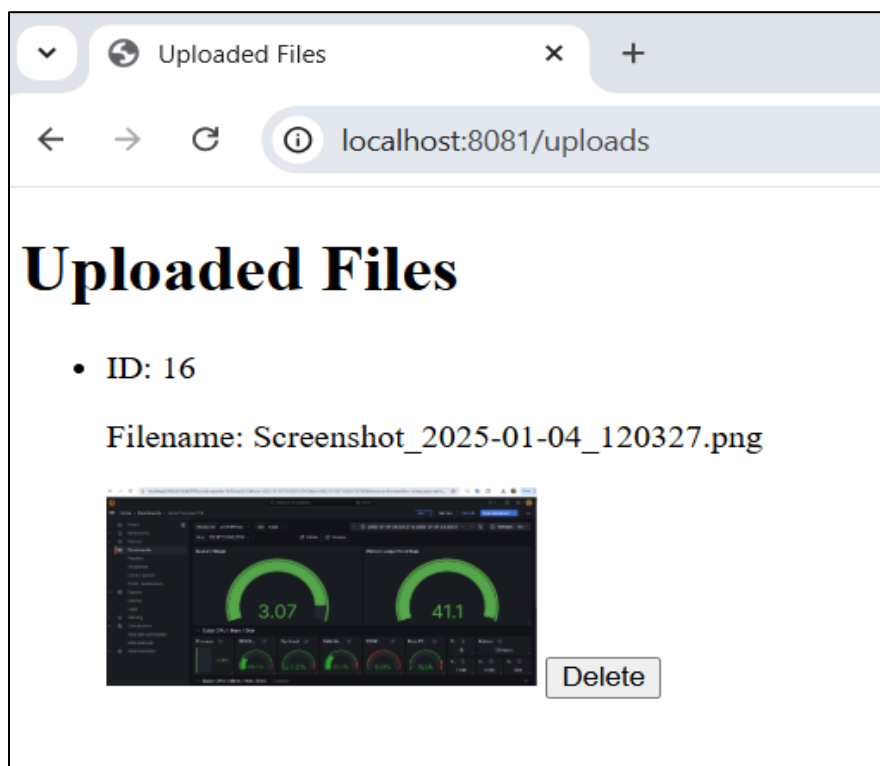
Tampilan frontend



Tampilan upload file



Tampilan sukses upload



Tampilan list upload

```
linux_2@T3S4PHS8:~/monitoring/server-agent/crud$ docker exec -it postgres psql -U myuser -d mydatabase
psql (17.2 (Debian 17.2-1.pgdg120+1))
Type "help" for help.

mydatabase=# SELECT * FROM file_upload;
 id |          filename          |
-----+-----
  16 | Screenshot_2025-01-04_120327.png |
(1 row)

mydatabase=# |
```

Tampilan postgres