Tugas Peserta:

1. Buat Docker Compose untuk menjalankan 2 service (backend & database mysql)
2. Gunakan Docker Volume untuk menyimpan data dari database mysql.
3. Pastikan container saling terhubung melalui jaringan.

Konsep singkat dulu

- **Docker Compose** → menjalankan banyak container sekaligus
- **Service 1** → Backend (misal Node.js / Python / dll)
- **Service 2** → Database MySQL
- **Docker Volume** → supaya data MySQL **tidak hilang** walau container mati
- **Network** → otomatis dari Docker Compose (service bisa saling akses pakai nama service)

Struktur folder

```
project/
│
├── docker-compose.yml
└── backend/
    ├── Dockerfile
    ├── package.json
    └── index.js
```

1. SSH to the instance

```
PS C:\Users\Valin\Downloads> ssh -i valin.pem ubuntu@47.129.4.12
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

2. Install docker

```
ubuntu@ip-172-31-28-183:~$ history
    1  sudo apt update
    2  sudo apt install -y docker.io
```

3. Aktifkan docker

```
ubuntu@ip-172-31-28-183:~$ sudo systemctl start docker
ubuntu@ip-172-31-28-183:~$ sudo systemctl enable docker
```

4. Install docker compose

```
ubuntu@ip-172-31-28-183:~$ sudo apt install -y docker-compose
```

5. Lakukan pengecekan docker dan docker compose

```
ubuntu@ip-172-31-28-183:~$ docker --version
Docker version 28.2.2, build 28.2.2-0ubuntu1~24.04.1
ubuntu@ip-172-31-28-183:~$ docker-compose --version
docker-compose version 1.29.2, build unknown
```

6. Ijinkan user jalankan docker

```
ubuntu@ip-172-31-28-183:~$ sudo usermod -aG docker ubuntu
ubuntu@ip-172-31-28-183:~$ exit
logout
Connection to 47.129.4.12 closed.
```

7. Login again

```
PS C:\Users\Valin\Downloads> ssh -i valin.pem ubuntu@47.129.4.12
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

8. Buat struktur project

```
 11  mkdir docker-app
 12  cd docker-app
 13  mkdir backend
```

cd backend

9. Buat backend sederhana (di directory backend)

```
14  nano Dockerfile
```

```
ubuntu@ip-172-31-28-183:~/docker-app/backend$ cat Dockerfile
FROM node:18

WORKDIR /app

COPY package.json .
RUN npm install

COPY . .

EXPOSE 3000
CMD ["node", "index.js"]
```

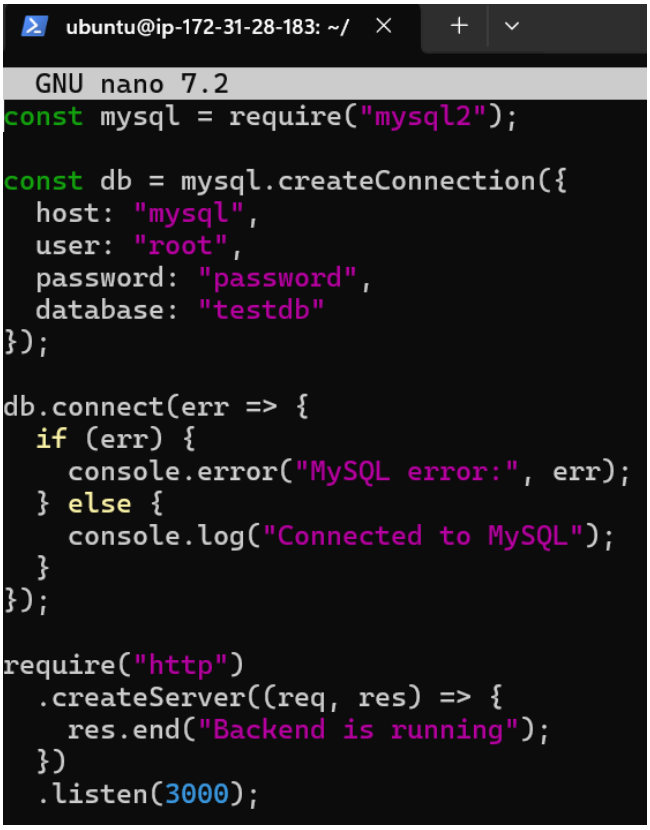
10. Buat package.json

```
ubuntu@ip-172-31-28-183:~/docker-app/backend$ cat package.json
{
  "name": "backend-app",
  "version": "1.0.0",
  "main": "index.js",
  "dependencies": {
    "express": "^4.18.2",
    "mysql2": "^3.6.0"
  }
}
```

11. Buat index.js

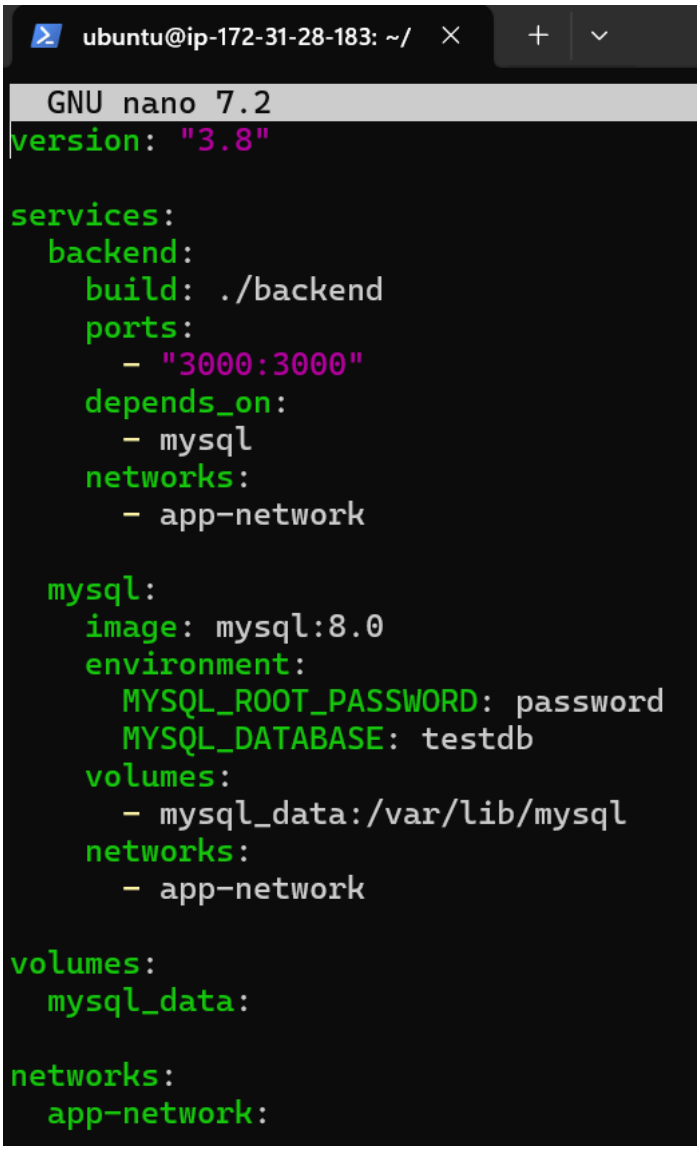
```
ubuntu@ip-172-31-28-183:~/docker-app/backend$ cat index.js
const express = require("express");
const mysql = require("mysql2");

const app = express();

const db = mysql.createConnection({
  host: process.env.DB_HOST,
  user: process.env.DB_USER,
  password: process.env.DB_PASSWORD,
  database: process.env.DB_NAME,
});

db.connect(err => {
  if (err) {
    console.error("DB connection failed:", err);
    return;
  }
  console.log("Connected to MySQL!");
});

app.get("/", (req, res) => {
  res.send("Backend connected to MySQL 🚀");
});

app.listen(3000, () => {
  console.log("Backend running on port 3000");
});
```

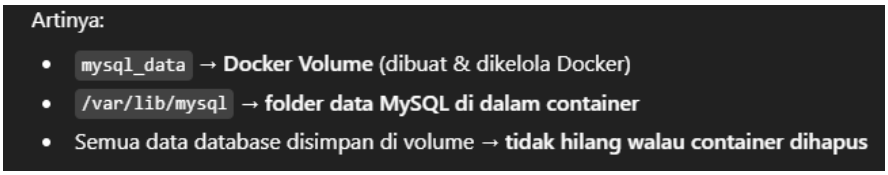
```
15  nano app.js
```



12. Buat docker compose – di docker compose ini, docker volume digunakan

```
ubuntu@ip-172-31-28-183:~/docker-app$ nano docker-compose.yml
```





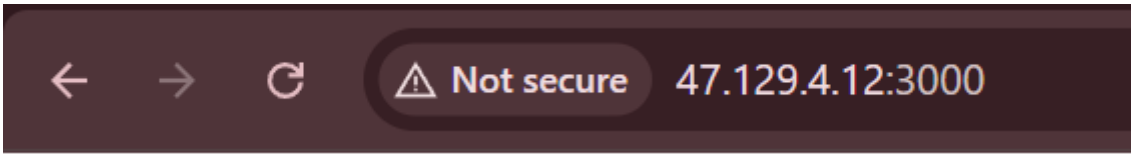
13. Jalankan container

```
ubuntu@ip-172-31-28-183:~/docker-app$ docker-compose up -d
Creating network "docker-app_app-network" with the default driver
Creating volume "docker-app_mysql_data" with default driver
Pulling mysql (mysql:8.0)...
```

14. Lakukan pengecekan

```
ubuntu@ip-172-31-28-183:~/docker-app/backend$ docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED          STATUS          PORTS                  NAMES
07d81c0175db   mysql:8.0   "docker-entrypoint.s…"   22 seconds ago   Up 22 seconds   3306/tcp, 33060/tcp   docker-app_my
sql_1
ubuntu@ip-172-31-28-183:~/docker-app/backend$ docker volume ls
DRIVER    VOLUME NAME
local     docker-app_mysql_data
ubuntu@ip-172-31-28-183:~/docker-app/backend$ docker network ls
NETWORK ID     NAME                     DRIVER    SCOPE
3319c2726627   bridge                   bridge    local
9c0e58b398e7   docker-app_app-network   bridge    local
5092b04cb9d3   host                     host      local
011cd7b59a30   none                     null      local
```

15. Check on the browser



Backend connected to MySQL 🚀