# Panduan Praktik Instalasi MongoDB dan Mongo Express GUI dengan Docker

## 📚 Pendahuluan

Panduan ini dirancang khusus untuk mahasiswa yang ingin mempelajari MongoDB melalui antarmuka grafis (GUI) menggunakan Mongo Express dengan Docker. Docker memungkinkan kita menjalankan database dan antarmuka pengguna dalam lingkungan yang terisolasi, konsisten, dan mudah diatur tanpa perlu instalasi yang kompleks di komputer lokal.

MongoDB adalah database NoSQL berbasis dokumen yang menyimpan data dalam format JSON-like, sementara Mongo Express menyediakan antarmuka web yang user-friendly untuk mengelola database MongoDB. Kombinasi ini sangat ideal untuk pembelajaran karena memungkinkan mahasiswa fokus pada konsep database tanpa terhalang oleh setup yang rumit.

## ✅ Prasyarat Sebelum Memulai

Sebelum memulai instalasi, pastikan komputer Anda telah terinstall dengan software berikut:

### 1. Docker Engine

Docker Engine diperlukan untuk menjalankan kontainer. Untuk memeriksa instalasi Docker, buka terminal dan jalankan:

docker --version

Output yang diharapkan:

Docker version 24.0.7, build afdd53b

### 2. Docker Compose

Docker Compose digunakan untuk mendefinisikan dan menjalankan aplikasi multi-kontainer. Periksa instalasi dengan:

docker compose version

Output yang diharapkan:

Docker Compose version v2.20.2

**Catatan:** Setup ini bekerja di Windows, macOS, dan Linux (Debian atau Ubuntu)

## 🚀 Langkah 1: Membuat Direktori Proyek

Buat direktori baru untuk menyimpan semua file konfigurasi:

mkdir mongo-docker-praktik  
cd mongo-docker-praktik

Direktori ini akan menjadi workspace untuk praktik MongoDB kita.

## 📝 Langkah 2: Membuat File docker-compose.yml

Langkah inti adalah membuat file konfigurasi docker-compose.yml. Buat file baru dengan nama tersebut di dalam direktori mongo-docker-praktik dan salin konfigurasi berikut:

version: '3.8'  
services:  
 mongo:  
 image: mongo:latest  
 container\_name: mongo-container  
 environment:  
 MONGO\_INITDB\_ROOT\_USERNAME: admin  
 MONGO\_INITDB\_ROOT\_PASSWORD: password  
 ports:  
 - "27017:27017"  
 volumes:  
 - mongo-data:/data/db  
 networks:  
 - mongo\_network  
  
 mongo-express:  
 image: mongo-express:latest  
 container\_name: mongo-express-container  
 ports:  
 - "8081:8081"  
 environment:  
 ME\_CONFIG\_MONGODB\_ADMINUSERNAME: admin  
 ME\_CONFIG\_MONGODB\_ADMINPASSWORD: password  
 ME\_CONFIG\_MONGODB\_SERVER: mongo  
 networks:  
 - mongo\_network  
  
networks:  
 mongo\_network:  
 driver: bridge  
  
volumes:  
 mongo-data:

### Penjelasan Konfigurasi:

#### Layanan MongoDB

* **image: mongo:latest**: Menggunakan image MongoDB resmi dari Docker Hub
* **container\_name**: Memberi nama kontainer agar mudah diidentifikasi
* **environment**: Mengatur username dan password admin
* **ports**: Mapping port 27017 untuk akses dari host
* **volumes**: Menyimpan data secara persisten menggunakan volume bernama mongo-data.

#### Layanan Mongo Express

* **image: mongo-express:latest**: Menggunakan image Mongo Express resmi
* **ports**: Mapping port 8081 untuk akses web interface
* **environment**: Konfigurasi koneksi ke MongoDB

#### Jaringan dan Volume

* **networks**: Membuat jaringan internal untuk komunikasi antar kontainer
* **volumes**: Mendeklarasikan volume untuk penyimpanan data persisten

## ⚡ Langkah 3: Menjalankan Aplikasi

Setelah file docker-compose.yml dibuat, jalankan perintah berikut untuk memulai semua layanan:

docker compose up -d

Perintah ini akan:

1. Mendownload image yang diperlukan (jika belum ada)
2. Membuat volume dan jaringan
3. Membuat dan menjalankan kontainer
4. Menjalankan kontainer di background mode (detach mode)

Proses ini mungkin memakan waktu beberapa menit tergantung kecepatan internet dan komputer Anda.

## 🔍 Langkah 4: Verifikasi Status Kontainer

Periksa apakah kontainer berjalan dengan baik:

docker ps

Output yang diharapkan:

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
xxxxxxxxxxxx mongo:latest "docker-entrypoint.s…" 2 minutes ago Up 2 minutes 0.0.0.0:27017->27017/tcp mongo-container  
xxxxxxxxxxxx mongo-express:latest "tini -- /docker-entr…" 2 minutes ago Up 2 minutes 0.0.0.0:8081->8081/tcp mongo-express-container

Pastikan kedua kontainer menunjukkan status Up.

### Memeriksa Log (Jika Ada Masalah)

Jika kontainer tidak berjalan dengan baik, periksa log untuk troubleshooting:

# Log MongoDB  
docker logs mongo-container  
  
# Log Mongo Express  
docker logs mongo-express-container

Log yang berhasil akan menampilkan pesan seperti [initandlisten] waiting for connections on port 27017 untuk MongoDB dan Mongo Express server listening at 8081 untuk Mongo Express [(netcup Community, 2021)](https://community.netcup.com/en/tutorials/how-to-install-mongodb-and-mongo-express-with-docker-and-connect-mongo-express-with-mongodb).

## 🌐 Langkah 5: Mengakses Mongo Express GUI

Buka browser web dan kunjungi URL berikut:

http://localhost:8081

Halaman login Mongo Express akan muncul. Masukkan kredensial berikut:

* **Username**: admin
* **Password**: password

Setelah berhasil login, Anda akan melihat dashboard utama Mongo Express yang menampilkan semua database yang tersedia.

## 💾 Langkah 6: Membuat Database dan Koleksi Pertama

Mari kita buat database sederhana untuk praktik:

### 1. Membuat Database Baru

1. Di dashboard Mongo Express, klik tombol **“Create Database”**
2. Masukkan nama database: db\_mahasiswa
3. Masukkan nama koleksi (collection): mahasiswa
4. Klik **“Create”**

### 2. Menambahkan Dokumen Pertama

1. Klik database db\_mahasiswa
2. Klik koleksi mahasiswa
3. Klik tombol **“Add Data”**
4. Masukkan data dalam format JSON:

{  
 "nim": "2021001",  
 "nama": "Ahmad Rizki",  
 "jurusan": "Teknik Informatika",  
 "semester": 6,  
 "mata\_kuliah": ["Database", "Pemrograman Web", "AI"],  
 "alamat": {  
 "jalan": "Jl. Merdeka No. 123",  
 "kota": "Jakarta",  
 "kode\_pos": "12345"  
 },  
 "ipk": 3.75  
}

1. Klik **“Save”**

![Tampilan antarmuka Mongo Express dengan database dan koleksi yang telah dibuat](data:image/png;base64;base64,)

Tampilan antarmuka Mongo Express dengan database dan koleksi yang telah dibuat

## 📊 Langkah 7: Operasi CRUD Dasar

Setelah database terbuat, mari kita praktikkan operasi dasar CRUD (Create, Read, Update, Delete).

### Create: Menambah Data Baru

Tambahkan beberapa dokumen lagi dengan data berbeda:

{  
 "nim": "2021002",  
 "nama": "Siti Nurhaliza",  
 "jurusan": "Sistem Informasi",  
 "semester": 4,  
 "mata\_kuliah": ["Basis Data", "Jaringan Komputer"],  
 "alamat": {  
 "jalan": "Jl. Sudirman No. 456",  
 "kota": "Bandung",  
 "kode\_pos": "40123"  
 },  
 "ipk": 3.85  
}

### Read: Membaca/Mencari Data

Mongo Express menyediakan fitur pencarian yang mudah:

1. **Menampilkan semua data**: Klik koleksi mahasiswa
2. **Mencari dengan kriteria spesifik**: Gunakan field pencarian dengan format JSON

Contoh pencarian mahasiswa dengan jurusan “Teknik Informatika”:

{"jurusan": "Teknik Informatika"}

### Update: Memperbarui Data

Untuk mengupdate data:

1. Temukan dokumen yang ingin diupdate
2. Klik tombol **“Edit”**
3. Ubah data yang diperlukan
4. Klik **“Save”**

Contoh update IPK mahasiswa:

{  
 "nim": "2021001",  
 "nama": "Ahmad Rizki",  
 "jurusan": "Teknik Informatika",  
 "semester": 6,  
 "mata\_kuliah": ["Database", "Pemrograman Web", "AI", "Machine Learning"],  
 "alamat": {  
 "jalan": "Jl. Merdeka No. 123",  
 "kota": "Jakarta",  
 "kode\_pos": "12345"  
 },  
 "ipk": 3.80  
}

### Delete: Menghapus Data

Untuk menghapus dokumen:

1. Temukan dokumen yang ingin dihapus
2. Klik tombol **“Delete”**
3. Konfirmasi penghapusan

## 🔧 Langkah 8: Query Lanjutan dengan Bagian Query

Mongo Express juga memiliki fitur untuk menjalankan query MongoDB langsung:

1. Klik database db\_mahasiswa
2. Klik tab **“Query”**
3. Masukkan query MongoDB dalam format JSON

### Contoh Query Lanjutan:

**Mencari mahasiswa dengan IPK > 3.8:**

{"ipk": {"$gt": 3.8}}

**Mencari mahasiswa dari Jakarta atau Bandung:**

{"alamat.kota": {"$in": ["Jakarta", "Bandung"]}}

**Menghitung jumlah mahasiswa per jurusan:**

[  
 {"$group": {"\_id": "$jurusan", "jumlah": {"$sum": 1}}}  
]

## 📋 Tabel Referensi Operasi MongoDB

| Operasi | SQL Equivalent | MongoDB Query | Mongo Express GUI |
| --- | --- | --- | --- |
| SELECT \* FROM table | db.collection.find() | Klik koleksi, view all | ✅ |
| WHERE condition | db.collection.find({field: value}) | Filter di interface | ✅ |
| INSERT INTO | db.collection.insertOne() | Add Data button | ✅ |
| UPDATE SET | db.collection.updateOne() | Edit button | ✅ |
| DELETE FROM | db.collection.deleteOne() | Delete button | ✅ |
| ORDER BY | db.collection.find().sort() | Query tab dengan sort | ⚠️ |
| GROUP BY | db.collection.aggregate() | Query tab dengan aggregate | ⚠️ |

*Legend: ✅ = Mudah di GUI, ⚠️ = Perlu query manual*

## 🛠️ Langkah 9: Menghubungkan Aplikasi Eksternal

Port 27017 telah di-expose, memungkinkan aplikasi eksternal terhubung ke MongoDB:

### Connection String untuk Node.js:

const mongoose = require('mongoose');  
mongoose.connect('mongodb://admin:password@localhost:27017/db\_mahasiswa?authSource=admin');

### Connection String untuk Python:

from pymongo import MongoClient  
client = MongoClient('mongodb://admin:password@localhost:27017/', authSource='admin')  
db = client['db\_mahasiswa']

## 🧹 Langkah 10: Menghentikan dan Membersihkan

Setelah selesai praktik, hentikan kontainer dengan:

docker compose down

Perintah ini akan:

* Menghentikan kontainer
* Menghapus kontainer dan jaringan
* **Mempertahankan data** di volume mongo-data

### Membersihkan Data (Jika Ingin Reset Lengkap):

docker compose down -v

Perintah ini menghapus semua data, berguna untuk memulai praktik dari awal.

## 🚨 Troubleshooting Umum

| Masalah | Penyebab | Solusi |
| --- | --- | --- |
| Cannot connect to MongoDB | Kontainer belum berjalan | Jalankan docker ps dan pastikan status Up |
| Connection refused di browser | Port 8081 belum ready | Tunggu 1-2 menit, refresh browser |
| Login gagal | Username/password salah | Periksa file docker-compose.yml |
| Data hilang setelah restart | Volume tidak terbuat | Pastikan volume mongo-data ada dengan docker volume ls |
| Port conflict | Port 27017/8081 digunakan | Ubah port mapping di docker-compose.yml |

### Memeriksa Volume Data:

# Melihat semua volume  
docker volume ls  
  
# Inspect volume mongo-data  
docker volume inspect mongo-data

## 🎯 Project Mini: Sistem Informasi Mahasiswa

Sebagai latihan akhir, coba buat struktur data berikut:

### Database: siakam

### Koleksi: mahasiswa

{  
 "nim": "2023001",  
 "nama\_depan": "Budi",  
 "nama\_belakang": "Santoso",  
 "tanggal\_lahir": "2002-03-15",  
 "email": "budi.santoso@universitas.ac.id",  
 "program\_studi": {  
 "kode": "TI",  
 "nama": "Teknik Informatika",  
 "fakultas": "FTI"  
 },  
 "semester": 4,  
 "status": "Aktif",  
 "mata\_kuliah\_diambil": [  
 {  
 "kode\_mk": "IF301",  
 "nama\_mk": "Basis Data",  
 "sks": 3,  
 "nilai": "A"  
 },  
 {  
 "kode\_mk": "IF302",  
 "nama\_mk": "Struktur Data",  
 "sks": 4,  
 "nilai": "B+"  
 }  
 ],  
 "prestasi": ["Juara 2 Hackathon 2023", "Asisten Lab Basis Data"],  
 "create\_at": "2023-09-01T10:00:00Z",  
 "update\_at": "2024-01-15T14:30:00Z"  
}

### 

### Tugas:

1. Insert 5 data mahasiswa dengan jurusan berbeda
2. Cari mahasiswa dengan IPK > 3.5
3. Update status mahasiswa menjadi “Lulus”
4. Hapus mahasiswa dengan nim tertentu
5. Hitung jumlah mahasiswa per program studi