

TUGAS
PRATIUM PEMOGRAM JARINGAN
“MongoDB”

Dosen Pengampu: Ade Kurniawan, S.Pd., M.Pd.T



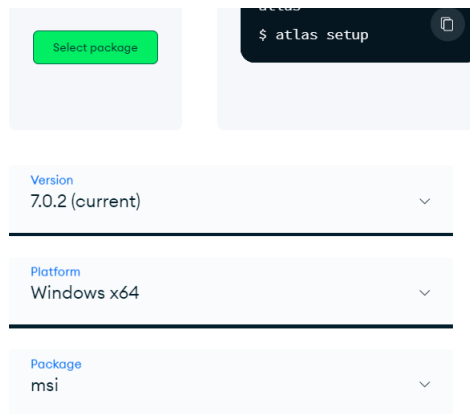
Disusun Oleh:
Najwa Alawiyah Siregar
(22346040)

PRODI INFORMATIKA
DEPARTEMENT TEKNIK ELEKTRONIKA
FAKULTAS TEKNIK
UNIVERSITAS NEGERI PADANG
TAHUN AJARAN 2024

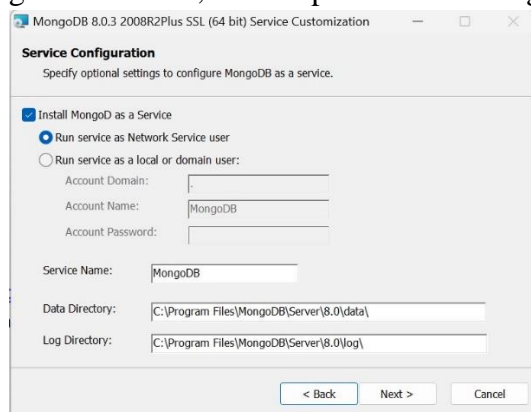
Latihan :

1. Instalasi MongoDB

- a. Silakan download **MongoDB Community Server** melalui link berikut ini <https://www.mongodb.com/try/download/community>
- b. Silakan klik select package hingga muncul seperti gambar berikut ini. Silakan pilih versi terbaru dan sesuaikan tipe sistem operasi anda. Pada **package**, pastikan anda memilih **msi** untuk memudahkan instalasi. Kemudian klik tombol **Download**

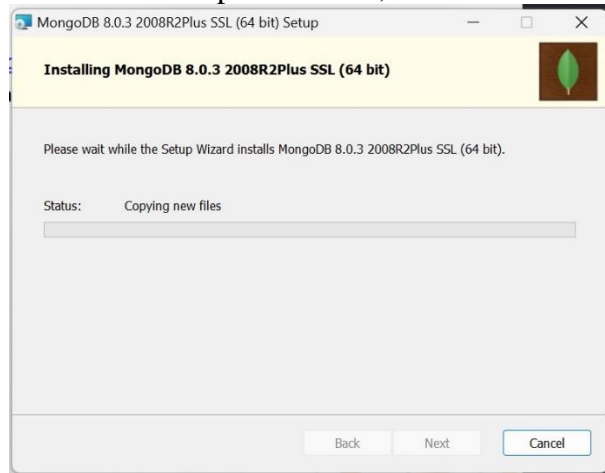


- c. Setelah selesai download, silakan lakukan instalasi. Jika muncul tampilan seperti gambar berikut, silakan pilih Install MongoDB as a service



Keterangan lebih lanjut terkait ini, bisa dibaca melalui link berikut <https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/>

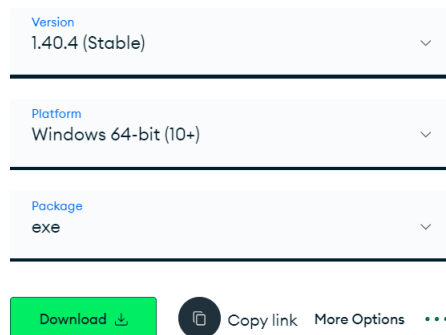
- d. Jika muncul tampilan berikut, silakan centan **Install MongoDB Compass**, lalu klik **Next**



- e. MongoDB Compass adalah tools berbasis GUI yang berguna untuk manajemen database MongoDB dengan mudah. Jika anda terlanjur melewati bagian tersebut d, silakan download MongoDB Compass melalui link berikut ini

<https://www.mongodb.com/try/download/compass>

- f. Pastikan anda mendownload versi terbaru dan memilih package **.exe**



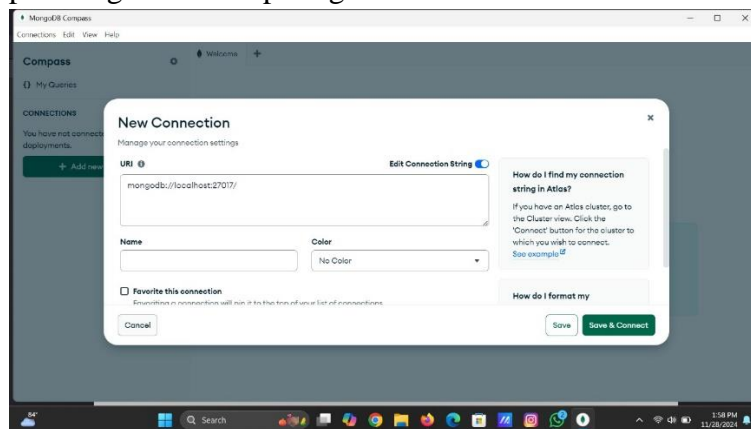
- g. Download dan lakukan proses instalasi hingga selesai
- h. Alternatif dari MongoDB Compass adalah **Studio 3T** dengan fitur yang lebih lengkap dan dapat didownload melalui link berikut <https://studio3t.com/>
- i. Untuk memastikan bahwa MongoDB server anda telah terinstall dengan baik dan telah berjalan, silakan buka **Windows Task Manager** (tekan **Ctrl + Shift + Esc**) dan perhatikanlah pada bagian **Processes** bahwa **MongoDB Database Server** telah berjalan.

- j. Jika belum berjalan, silakan klik **Services**, lalu carilah **MongoDB**, klik **Kanan**, lalu Pilih **Start**



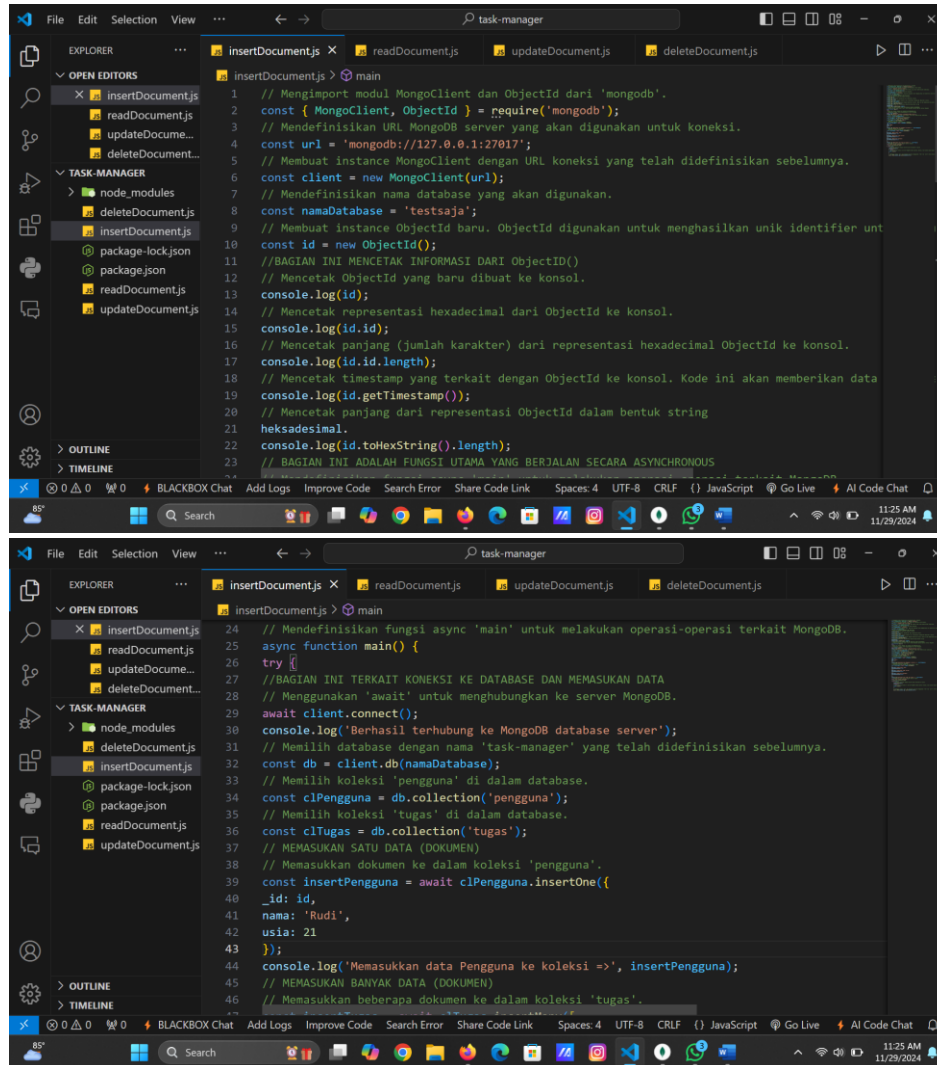
2. Koneksi ke database dan memasukan data dokumen (INSERT)

- a. **PENTING!** Semua dokumentasi dan penjelasan terkait perintah-perintah MongoDB yang akan digunakan dalam praktik ini telah tersedia melalui link berikut <https://mongodb.github.io/node-mongodb-native/6.2/>
- b. Bukalah aplikasi **MongoDB Compass** anda dan ketikkanlah **mongodb://localhost:27017** pada bagian **URI** seperti gambar dibawah ini. Lalu klik **Connect**



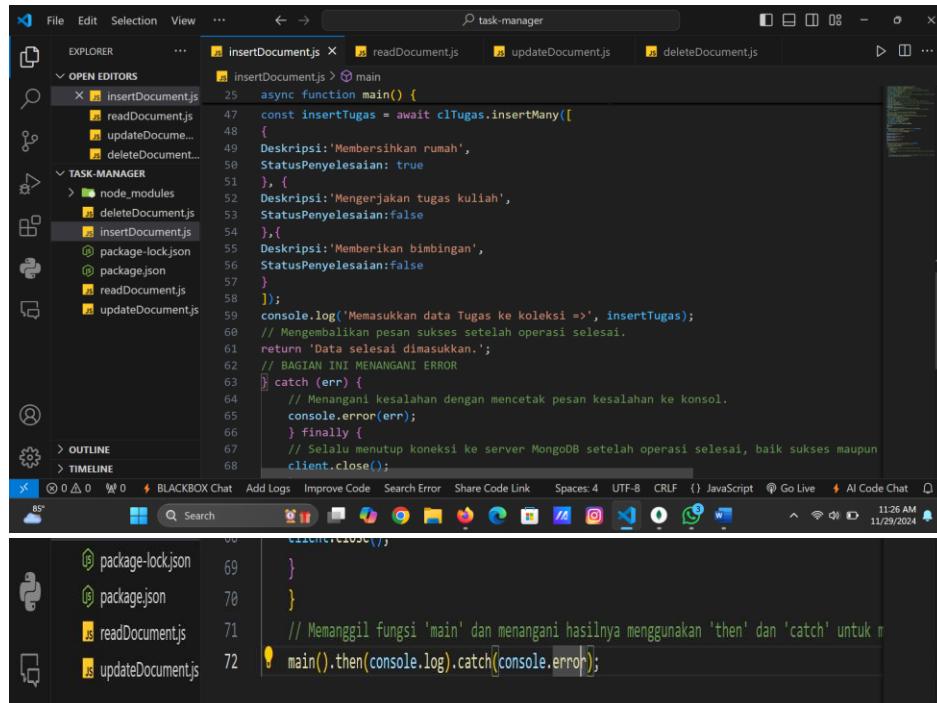
- c. Biarkan aplikasi tersebut tetap terbuka. Lalu, buatlah **folder baru** dengan nama **task-manager** pada visual studio code anda.
- d. Buka terminal pada visual studio code anda dan ketikkan **npm init -y** untuk meng-generate file package.json

- e. Lakukan **instalasi library** mongodb dengan perintah **npm i mongodb@6.2.0**. Berikut adalah link library mongodb <https://www.npmjs.com/package/mongodb>
- f. Setelah itu, buatlah **file JavaScript** baru dengan nama **insertDocument.js** dalam foldertask-manager
- g. Ketikkanlah *source code* berikut ini pada file **insertDocument.js** untuk melakukan koneksi dan memasukkan data (*insert*) ke database mongodb



```
1 // Mengimport modul MongoClient dan ObjectId dari 'mongodb'.
2 const { MongoClient, ObjectId } = require('mongodb');
3 // Mendefinisikan URL MongoDB server yang akan digunakan untuk koneksi.
4 const url = 'mongodb://127.0.0.1:27017';
5 // Membuat instance MongoClient dengan URL koneksi yang telah didefinisikan sebelumnya.
6 const client = new MongoClient(url);
7 // Mendefinisikan nama database yang akan digunakan.
8 const namaDatabase = 'testaja';
9 // Membuat instance ObjectId baru. ObjectId digunakan untuk menghasilkan unik identifier untuk
10 const id = new ObjectId();
11 // BAGIAN INI MENCETAK INFORMASI DARI ObjectId()
12 // Mencetak ObjectId yang baru dibuat ke konsol.
13 console.log(id);
14 // Mencetak representasi hexadecimal dari ObjectId ke konsol.
15 console.log(id.id);
16 // Mencetak panjang (jumlah karakter) dari representasi hexadecimal ObjectId ke konsol.
17 console.log(id.id.length);
18 // Mencetak timestamp yang terkait dengan ObjectId ke konsol. Kode ini akan memberikan data
19 console.log(id.getTimestamp());
20 // Mencetak panjang dari representasi ObjectId dalam bentuk string
21 console.log(id.toHexString().length);
22 // BAGIAN INI ADALAH FUNGSI UTAMA YANG BERJALAN SECARA ASYNCHRONOUS
23
```

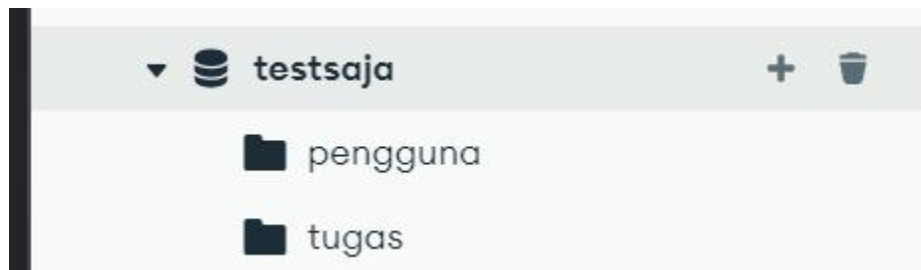
```
24 // Mendefinisikan fungsi async 'main' untuk melakukan operasi-operasi terkait MongoDB.
25 async function main() {
26   try {
27     // BAGIAN INI TERKAIT KONEKSI KE DATABASE DAN MEMASUKAN DATA
28     // Menggunakan 'await' untuk menghubungkan ke server MongoDB.
29     await client.connect();
30     console.log('Berhasil terhubung ke MongoDB database server');
31     // Memilih database dengan nama 'task-manager' yang telah didefinisikan sebelumnya.
32     const db = client.db(namaDatabase);
33     // Memilih koleksi 'pengguna' di dalam database.
34     const cIPengguna = db.collection('pengguna');
35     // Memilih koleksi 'tugas' di dalam database.
36     const cITugas = db.collection('tugas');
37     // MEMASUKAN SATU DATA (DOKUMEN)
38     // Memasukkan dokumen ke dalam koleksi 'pengguna'.
39     const insertPengguna = await cIPengguna.insertOne({
40       _id: id,
41       nama: 'Rudi',
42       usia: 21
43     });
44     console.log('Memasukkan data Pengguna ke koleksi =>', insertPengguna);
45     // MEMASUKAN BANYAK DATA (DOKUMEN)
46     // Memasukkan beberapa dokumen ke dalam koleksi 'tugas'.
47
```



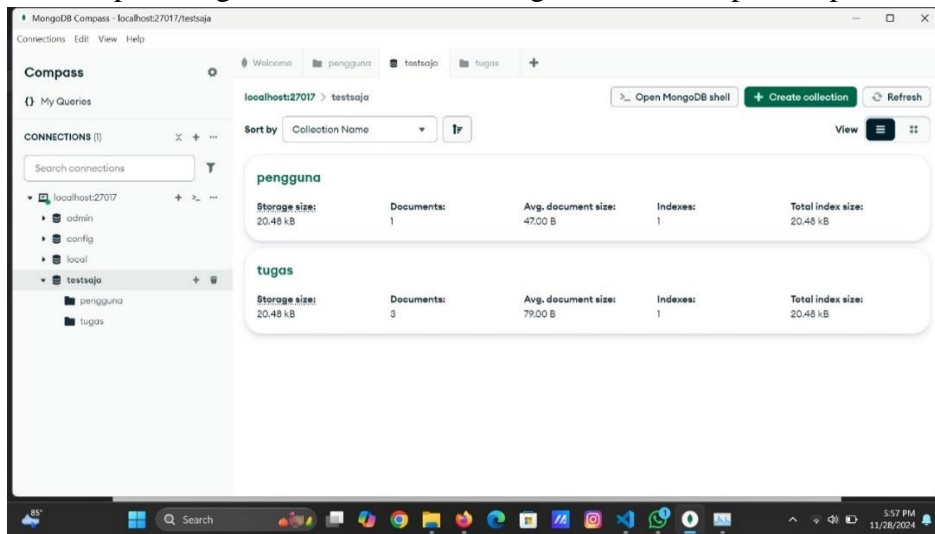
The screenshot shows the Visual Studio Code editor interface. The Explorer sidebar on the left displays the project structure for 'task-manager', including files like 'insertDocument.js', 'readDocument.js', 'updateDocument.js', and 'deleteDocument.js'. The main editor window is open to 'insertDocument.js', showing a JavaScript file with an async function 'main()' that uses 'mongoose' to insert multiple documents into a MongoDB collection. The code includes comments in Indonesian and error handling with 'try-catch' and 'console.error()'. The status bar at the bottom indicates the file is in 'JavaScript' mode.

```
25 async function main() {
47   const insertTugas = await cliTugas.insertMany([
48     {
49       Deskripsi: 'Membersihkan rumah',
50       StatusPenyelesaian: true
51     }, {
52       Deskripsi: 'Mengerjakan tugas kuliah',
53       StatusPenyelesaian: false
54     }, {
55       Deskripsi: 'Memberikan bimbingan',
56       StatusPenyelesaian: false
57     }
58   ]);
59   console.log('Memasukkan data Tugas ke koleksi =>', insertTugas);
60   // Mengembalikan pesan sukses setelah operasi selesai.
61   return 'Data selesai dimasukkan.';
62   // BAGIAN INI MENANGANI ERROR
63 } catch (err) {
64   // Menangani kesalahan dengan mencetak pesan kesalahan ke konsol.
65   console.error(err);
66 } finally {
67   // Selalu menutup koneksi ke server MongoDB setelah operasi selesai, baik sukses maupun
68   client.close();
69 }
70 }
71 // Memanggil fungsi 'main' dan menangani hasilnya menggunakan 'then' dan 'catch' untuk
72 main().then(console.log).catch(console.error);
```

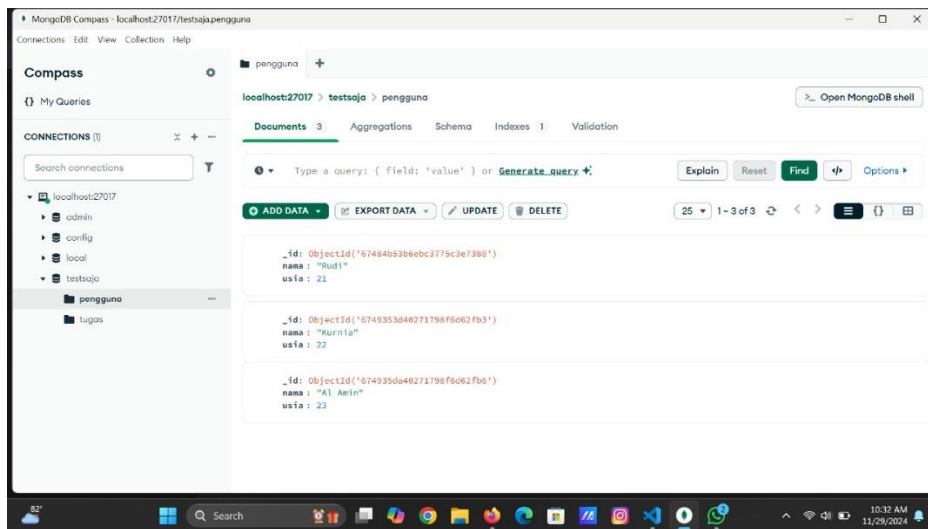
- h. Jalankan coding tersebut dengan mengetikan perintah **node insertDocument.js** dan **perhatikan** apa yang ditampilkan pada terminal
- i. Lalu, silakan buka **MongoDBCompass** anda dan perhatikanlah bahwa **database** terbaru dengan nama **task-manager** telah dibuat lengkap dengan **koleksi** yang diberi namapengguna dan tugas



- j. Anda dapat mengklik database tersebut agar muncul tampilan seperti berikut ini

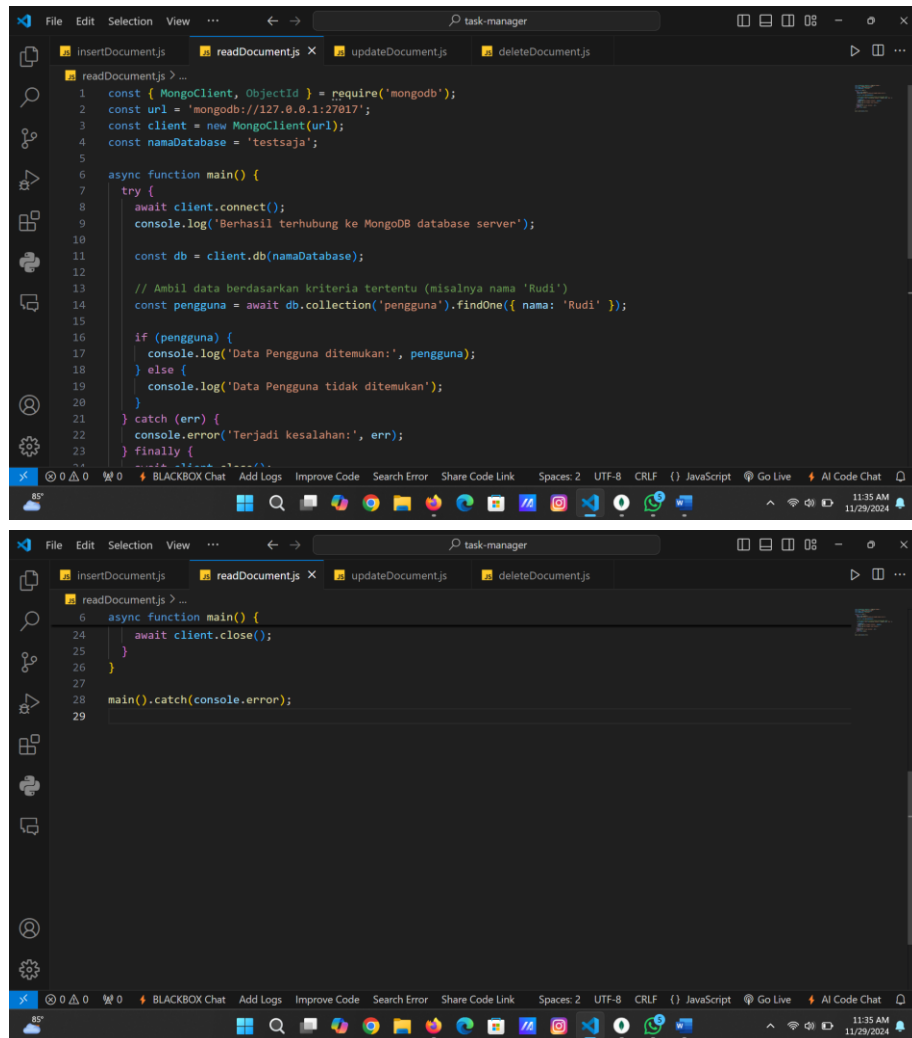


- k. Kliklah salah satu koleksi, misalnya 'pengguna' untuk melihat data yang telah anda masukan. Anda dapat mengubah tampilan data dengan mengklik **ikon yang diberi lingkaran merah** pada gambar dibawah ini. Pilihlah tampilan yang paling nyaman bagi anda.



3. Query dokumen (READ)

- a. Buatlah file JavaScript baru dengan nama **readDocument.js**, lalu masukanlah source code berikut ini

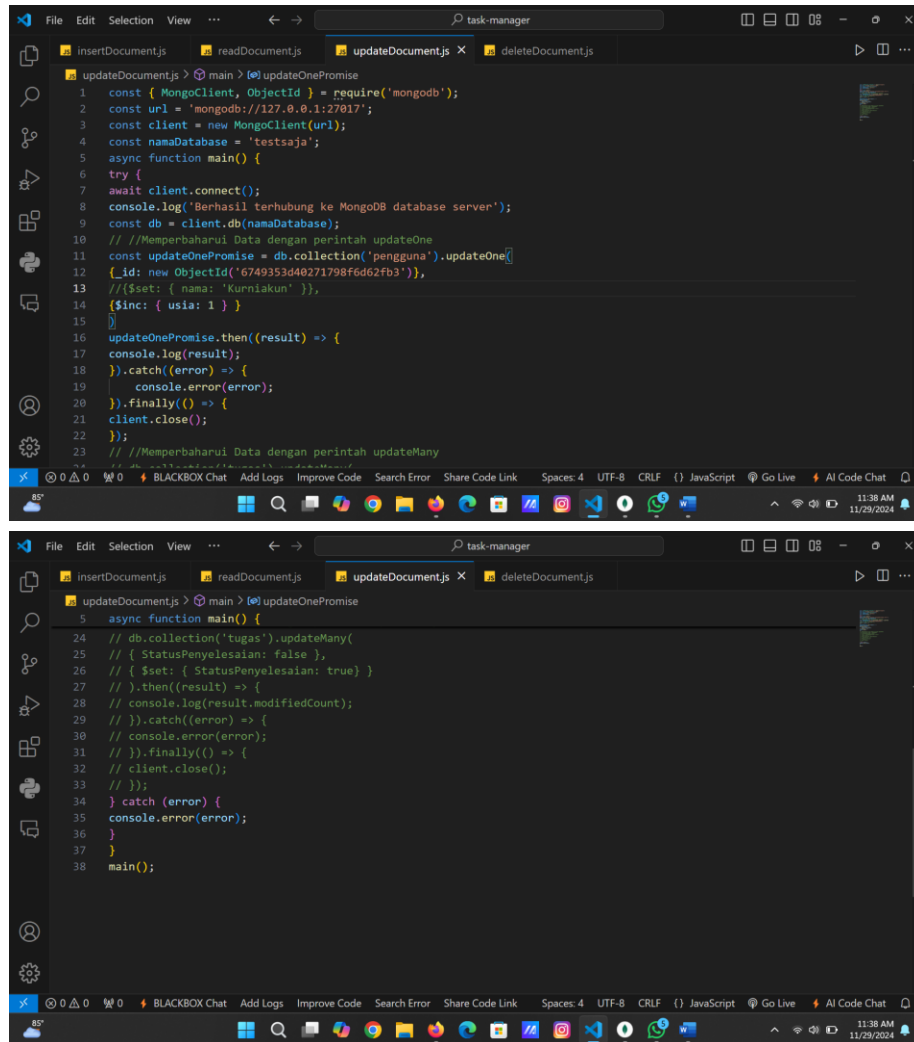


```
1 const { MongoClient, ObjectId } = require('mongodb');
2 const url = 'mongodb://127.0.0.1:27017';
3 const client = new MongoClient(url);
4 const namaDatabase = 'testsaja';
5
6 async function main() {
7   try {
8     await client.connect();
9     console.log('Berhasil terhubung ke MongoDB database server');
10
11     const db = client.db(namaDatabase);
12
13     // Ambil data berdasarkan kriteria tertentu (misalnya nama 'Rudi')
14     const pengguna = await db.collection('pengguna').findOne({ nama: 'Rudi' });
15
16     if (pengguna) {
17       console.log('Data Pengguna ditemukan:', pengguna);
18     } else {
19       console.log('Data Pengguna tidak ditemukan');
20     }
21   } catch (err) {
22     console.error('Terjadi kesalahan:', err);
23   } finally {
24     await client.close();
25   }
26 }
27
28 main().catch(console.error);
```

- a. Silakan ganti teks yang berwarna kuning dengan data **nama** yang ingin anda cari
- b. Ganti juga teks warna merah dengan **ID objek** yang ingin anda cari. ID objek ini dapat ditemukan pada data anda di MongoDB Compass. **Perhatikanlah** gambar pada tutorial sebelumnya terkait Insert Document bagian j
- c. Lalu, ganti juga teks warna biru dengan data **usia** yang ingin anda cari
- d. Jalankan kode diatas dengan perintah **node readDocument.js** dan **perhatikan** apa yang ditampilkan pada terminal

4. Memperbaharui Dokumen (UPDATE)

- a. Buatlah file baru dalam folder task-manager anda dan beri nama **updateDocument.js**, lalumasukanlah source code berikut ini

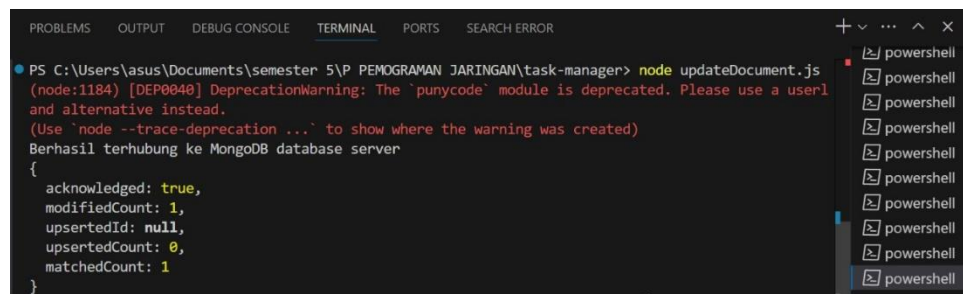


```
1 const { MongoClient, ObjectId } = require('mongodb');
2 const url = 'mongodb://127.0.0.1:27017';
3 const client = new MongoClient(url);
4 const namaDatabase = 'testsaja';
5 async function main() {
6   try {
7     await client.connect();
8     console.log('Berhasil terhubung ke MongoDB database server');
9     const db = client.db(namaDatabase);
10    // //Memperbaharui Data dengan perintah updateOne
11    const updateOnePromise = db.collection('pengguna').updateOne(
12      { _id: new ObjectId('6749353d40271798f6d62fb3') },
13      { $set: { nama: 'Kurniakun' } },
14      { $inc: { usia: 1 } }
15    );
16    updateOnePromise.then((result) => {
17      console.log(result);
18    }).catch((error) => {
19      console.error(error);
20    }).finally(() => {
21      client.close();
22    });
23    // //Memperbaharui Data dengan perintah updateMany
24    // db.collection('tugas').updateMany(
25    //   { StatusPenyelesaian: false },
26    //   { $set: { StatusPenyelesaian: true } }
27    // ).then((result) => {
28    //   console.log(result.modifiedCount);
29    // }).catch((error) => {
30    //   console.error(error);
31    // }).finally(() => {
32    //   client.close();
33    // });
34   } catch (error) {
35     console.error(error);
36   }
37 }
38 main();
```

- b. Gantilah **teks warna kuning** dengan salah satu **objectId** data anda dalam collection **pengguna**

- c. Gantilah **teks warna merah** dengan nama baru yang ingin anda perbaharui

Jalankan kode diatas dengan mengetikan perintah **node updateDocument.js** dan perhatikanlah bahwa terminal akan menampilkan pesan seperti gambar berikut ini.



```
PS C:\Users\asus\Documents\semester 5\P PEMOGRAMAN JARINGAN\task-manager> node updateDocument.js
(node:1184) [DEP0040] DeprecationWarning: The 'punycode' module is deprecated. Please use a userl
and alternative instead.
(Use 'node --trace-deprecation ...' to show where the warning was created)
Berhasil terhubung ke MongoDB database server
{
  acknowledged: true,
  modifiedCount: 1,
  upsertedId: null,
  upsertedCount: 0,
  matchedCount: 1
}
```

- d. **ModifiedCount** ditandai dengan **angka 1** menunjukkan bahwa 1 data telah diubah. Silakan cari tau makna dari masing-masing item yang lainnya (*acknowledged*, *upsertedId*, *upsertedCount* dan *matchedCount*) melalui internet untuk memperdalam pemahaman anda
- e. Setelah mengecek terminal anda, silakan buka juga aplikasi MongoDBCompass anda dan perhatikanlah bahwa data telah berubah
- f. Selanjutnya silakan jadikan baris kode berikut `{$set: { nama: 'Randikun' }}`, menjadi komentar. Lalu uncomment lah baris yang mengandung teks `$inc`.

```
_id: ObjectId('6749353d40271798f6d62fb3')
nama: "Kurniakun"
usia: 22
```

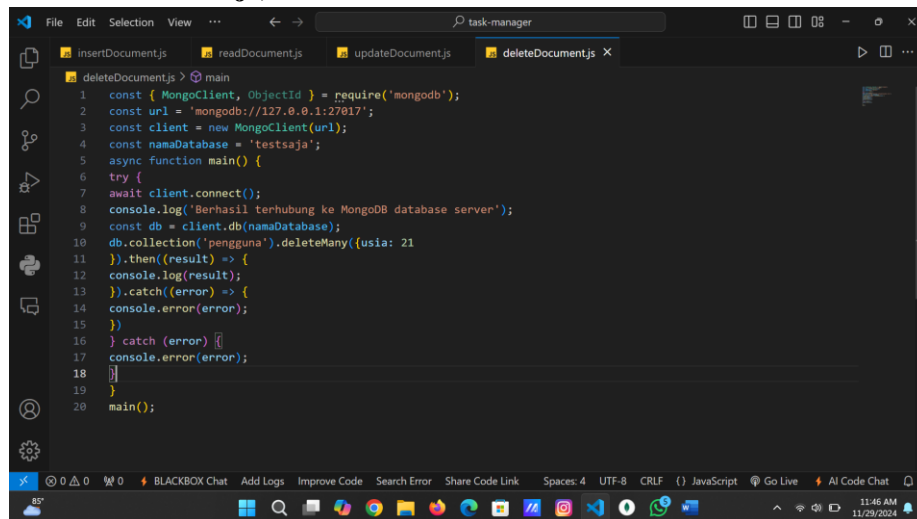
- g. Jalankan kembali file `updateDocument.js`, lalu perhatikan apa yang ditampilkan pada terminal dan hasilnya data usia telah berubah pada aplikasi MongoDBCompass.
- h. Selanjutnya silakan jadikan baris kode berikut `{$set: { nama: 'Randikun' }}`, menjadi komentar. Lalu uncomment lah baris yang mengandung teks `$inc`.

```
_id: ObjectId('6749353d40271798f6d62fb3')
nama: "Kurniakun"
usia: 23
```

- i. Jalankan kembali file `updateDocument.js`, lalu perhatikan apa yang ditampilkan pada terminal dan hasilnya data usia telah berubah pada aplikasi MongoDBCompass.

5. Menghapus dokumen (DELETE)

- a. Silakan buat file baru dalam folder `task-manager` anda dengan nama **`deleteDocument.js`**, lalu masukanlah kode berikut ini



```
1 const { MongoClient, ObjectId } = require('mongodb');
2 const url = 'mongodb://127.0.0.1:27017';
3 const client = new MongoClient(url);
4 const namaDatabase = 'testsaja';
5 async function main() {
6   try {
7     await client.connect();
8     console.log('Berhasil terhubung ke MongoDB database server');
9     const db = client.db(namaDatabase);
10    db.collection('pengguna').deleteMany({usia: 21
11  }).then(result => {
12    console.log(result);
13  }).catch(error => {
14    console.error(error);
15  })
16  } catch (error) {
17    console.error(error);
18  }
19 }
20 main();
```

- b. Jalankan kode tersebut dengan mengetikkan perintah **`node deleteDocument.js`** pada terminal
- c. Baris kode diatas digunakan untuk menghapus semua data **pengguna** yang **berusia**

22. Perhatikanlah apa yang ditampilkan pada terminal ketika anda menjalankan kode tersebut.

- d. Setelah itu, silakan cek data anda pada aplikasi **MongoDB Compass** dan perhatikanlah data apa yang telah dihapus. Jika anda masih bingung, silakan ganti angka **28** dengan data yang **paling banyak muncul** dalam data anda.

