

The background of the entire image is a dense, repeating pattern of stylized tropical leaves. The leaves are in various shades of green, yellow, and blue, creating a vibrant and naturalistic feel. In the center of the image is a large, solid blue circle. Inside this circle, the text 'mongoDB' is displayed. The word 'mongo' is in a dark brown, lowercase serif font, and 'DB' is in a light grey, uppercase serif font. A small, single green leaf is positioned to the left of the 'mongo' text, partially overlapping the blue circle. A faint, diagonal watermark reading 'babastudio.com' is visible across the blue circle.

mongoDB

babastudio.com

WEB CONSULTANT & TRAINING

BAB I

A. PERKENALAN DENGAN MONGO DB



MongoDB adalah cross - platform, database berorientasi dokumen yang menyediakan, kinerja tinggi, ketersediaan yang tinggi dan skalabilitas mudah. **MongoDB** bekerja pada

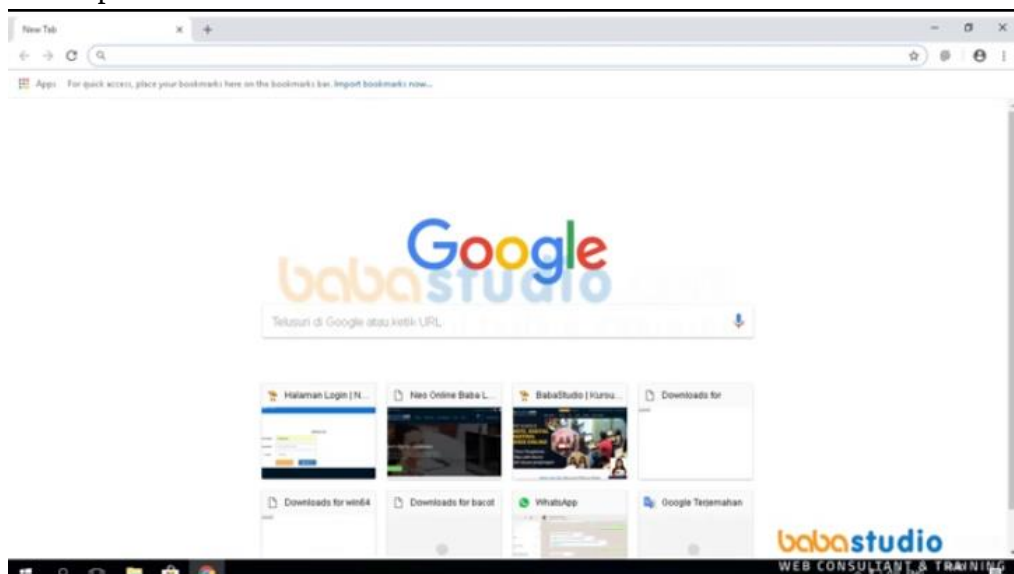
konsep Collection dan dokumen.

B. MENGAPA HARUS MENGGUNAKAN MONGO DB

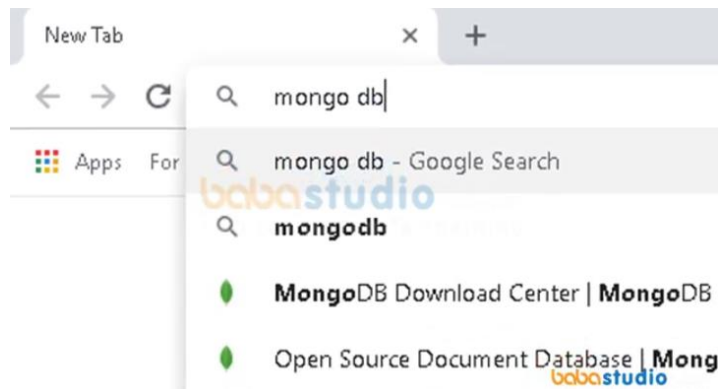
1. Penyimpanan berorientasi dokumen: Data disimpan dalam bentuk **JSON** gaya dokumen
2. Indeks pada atribut apapun
3. Replikasi & ketersediaan tinggi
4. *Auto-Sharding*
5. Kaya query
6. Update cepat di-tempat
7. Dukungan profesional oleh **MongoDB**

C. CARA DOWNLOAD MONGODB PADA WINDOWS

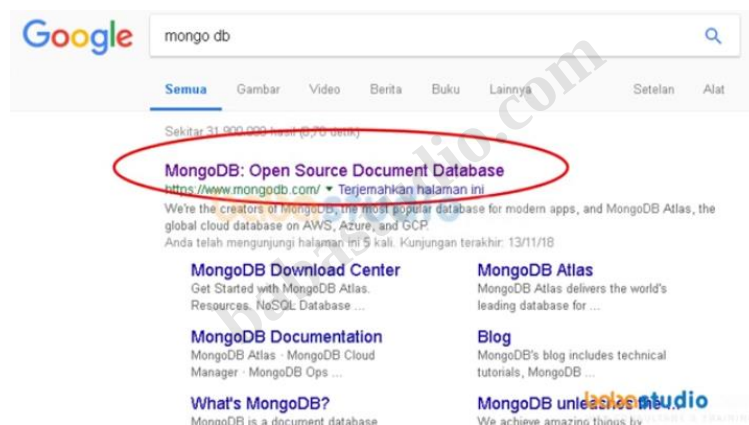
1. Buka aplikasi web browser terlebih dahulu



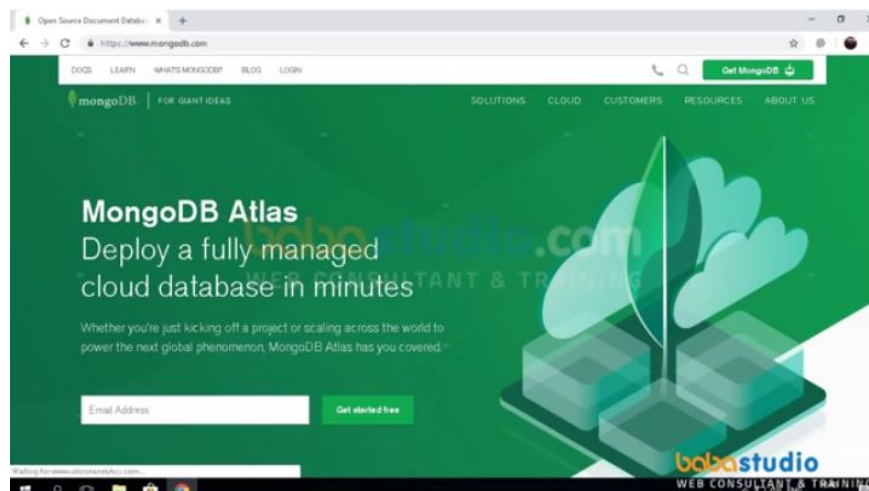
- Setelah dibuka ketik mongo db pada mesin pencaharian



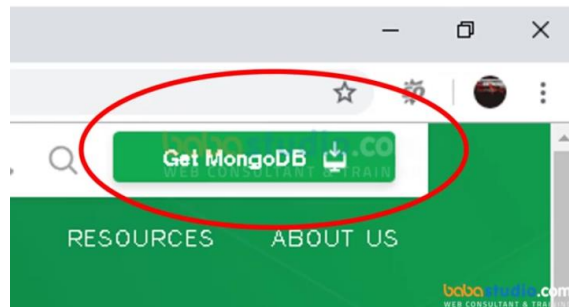
- Lalu akan tampil seperti gambar di bawah ini, kemudian pilih “MongoDB: Open Source Document Database”



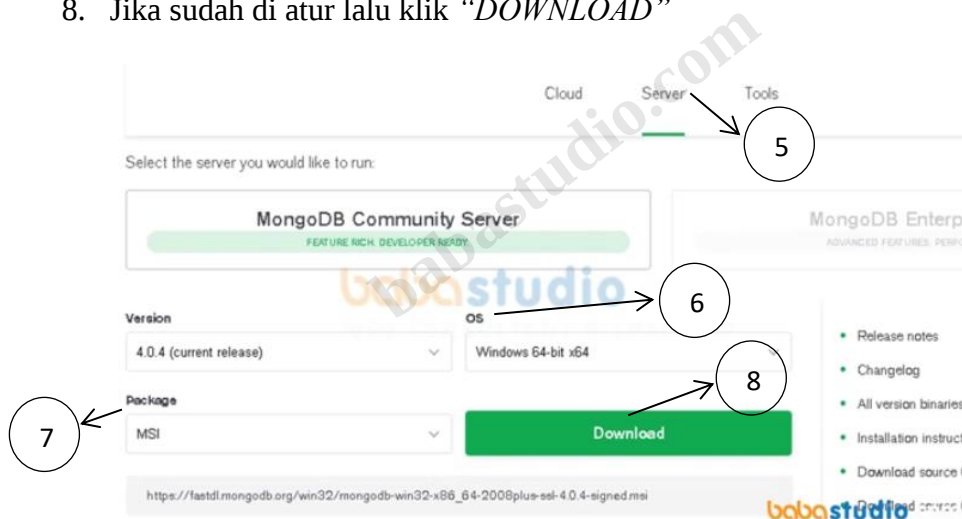
- Kemudian akan tampil gambar seperti ini



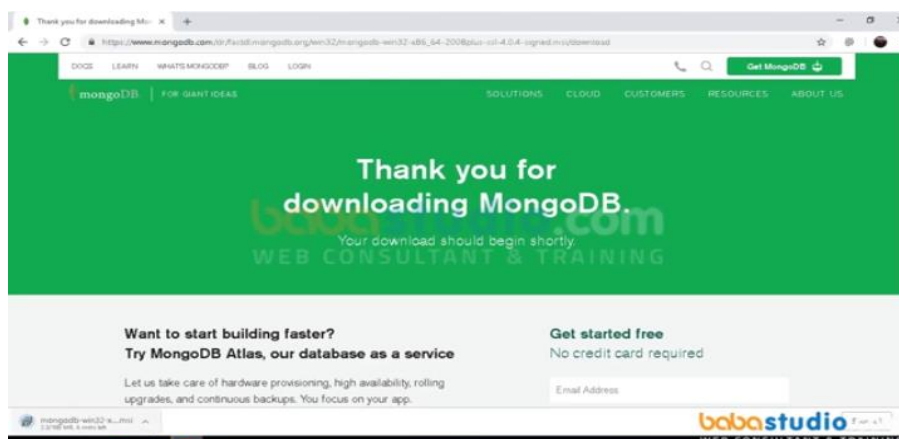
Setelah itu pilih “Get MongoDB” pada kotak sebelah kanan atas



5. Setelah di klik, arahkan mouse ke bawah lalu pilih “server”
6. Di sini terdapat pilihan “OS” yang bisa di instal, lalu kita pilih “windows 64 bit x64”
7. Kemudian atur kembali “PACKAGE” menggunakan “MSI”
8. Jika sudah di atur lalu klik “DOWNLOAD”



9. Lalu akan terotomatis terdownload



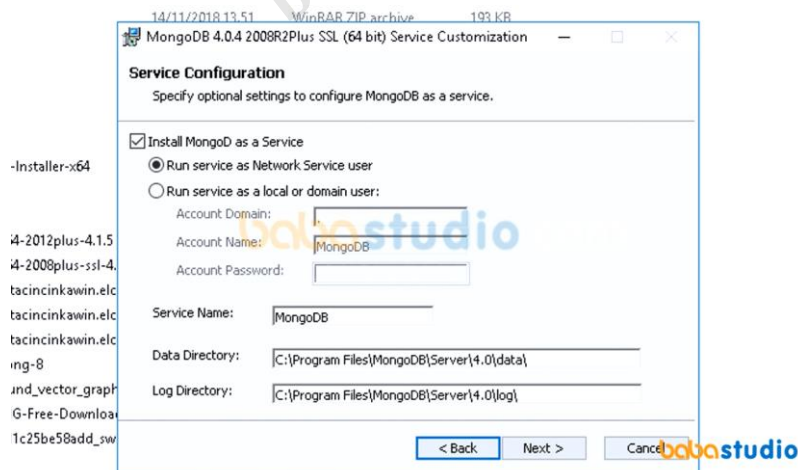
D. CARA MENGINSTAL MONGODB PADA WINDOWS

1. Klik 2x pada file mongoDB yang telah di download
2. Setelah di klik maka akan tampil seperti ini



Di klik next saja lalu ceklist *i accept the terms in the License Agreement* lalu klik next kembali.

3. Lalu pilih penginstalan Complete
4. Jika muncul seperti ini klik next saja

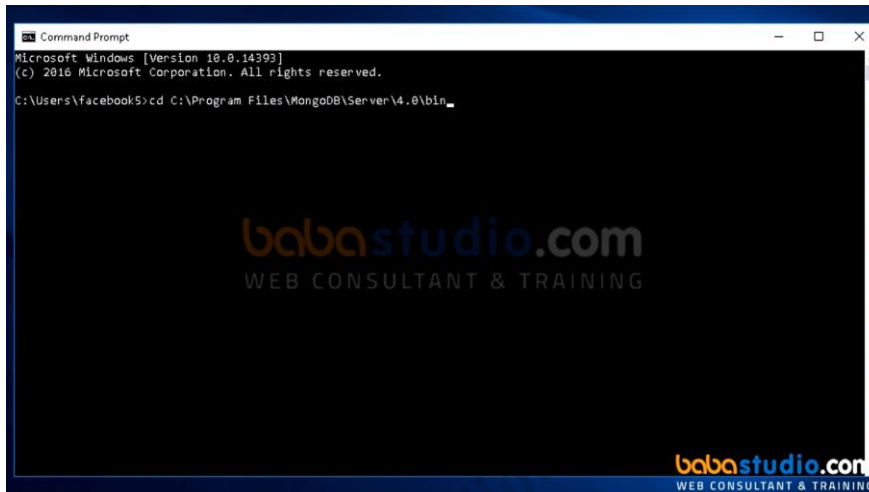


Lalu klik next lagi dan klik tombol install untuk menginstall mongoDB

5. Klik finish.

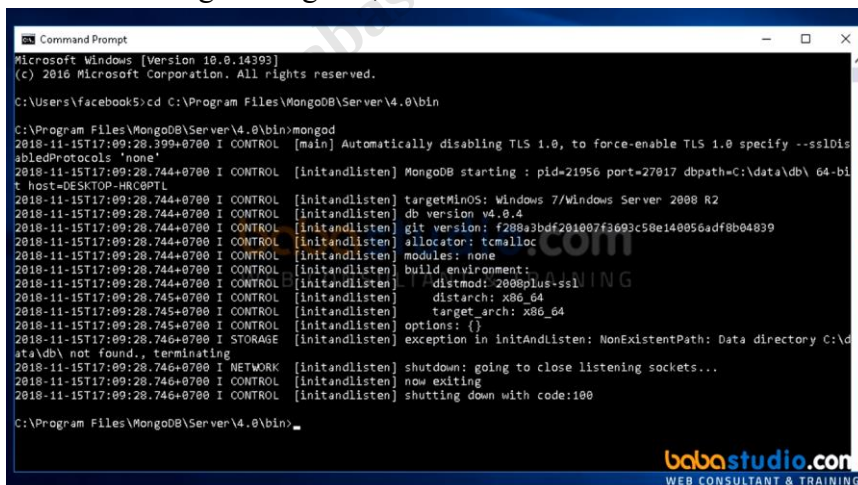
E. CARA MENGGUNAKAN MONGODB

1. Buka folder penginstalan mongodb (biasanya terdapat pada local disk C:/program files/mongoDB/server/4.0/bin) lalu copy directory folder penginstalan mongodb
2. Buka Command Prompt masing – masing dengan cara. Klik start > tulis pada pencarian CMD
3. Ketikan “cd C:/program files/mongoDB/server/4.0/bin (paste direktori yang sudah di copy”



Lalu tekan enter

4. Lalu ketikan lagi “mongod”, tekan enter



Maka tampilannya akan seperti ini

Sekarang coba perhatikan pada baris ke 5 dari bawah disitu ada bacaan *Data directory C:\data\db\ not found.*

coba sekarang buka foldernya ke localdisk C lalu kita buat folder baru bernama “data”, masuk ke folder data lalu kita buat folder lagi bernama “db”.

Lalu buka kembali Command Prompt nya lalu ketikan lagi “mongod”

```
ver.
2018-11-15T17:11:18.228+0700 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify
which IP
2018-11-15T17:11:18.228+0700 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --
bind_ip all to
2018-11-15T17:11:18.229+0700 I CONTROL [initandlisten] ** bind to all interfaces. If this behavior is desired,
start the
2018-11-15T17:11:18.229+0700 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warn
ing.
2018-11-15T17:11:18.229+0700 I CONTROL [initandlisten]
2018-11-15T17:11:18.238+0700 I STORAGE [initandlisten] createCollection: admin.system.version with provided UUID: 2cb54
37-90eb-4387-a930-859831471e1
2018-11-15T17:11:18.483+0700 I COMMAND [initandlisten] setting featureCompatibilityVersion to 4.0
2018-11-15T17:11:18.486+0700 I STORAGE [initandlisten] createCollection: local.startup_log with generated UUID: f43d8fc
c-9f4a-4d44-8cec-093f4f056e17
2018-11-15T17:11:18.757+0700 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory 'C
:/data/db/diagnostic.data'
2018-11-15T17:11:18.762+0700 I STORAGE [LogicalSessionCacheRefresh] createCollection: config.system.sessions with gener
ated UUID: 742339fc-6225-4b4b-adb4-41bc9f183da
2018-11-15T17:11:18.762+0700 I NETWORK [initandlisten] waiting for connections on port 27017
2018-11-15T17:11:18.976+0700 I INDEX [LogicalSessionCacheRefresh] build index on: config.system.sessions properties:
{ v: 2, key: { lastUse: 1 }, name: "lsidTTLIndex", ns: "config.system.sessions", expireAfterSeconds: 1800 }
2018-11-15T17:11:18.976+0700 I INDEX [LogicalSessionCacheRefresh] building index using bulk method; build may tem
porarily use up to 500 megabytes of RAM
2018-11-15T17:11:19.085+0700 I INDEX [LogicalSessionCacheRefresh] build index done. scanned 0 total records. 0 secs
2018-11-15T17:11:19.089+0700 I COMMAND [LogicalSessionCacheRefresh] command config.$cmd command: createIndexes { create
Indexes: [ "system.sessions", indexes: [ { key: { lastUse: 1 }, name: "lsidTTLIndex", expireAfterSeconds: 1800 } ], $db: "
config" } numFields:0 reslen:114 locks:{ Global: { acquireCount: { r: 2, w: 2 } }, Database: { acquireCount: { w: 2, w:
1 } }, Collection: { acquireCount: { w: 2 } } } protocol:op_msg 327ms
```

Maka tampilannya akan seperti ini.

5. Buka Command Prompt baru lalu lakukan kembali langkah ke 3 setelah itu coba di ketikan “mongo”, tekan enter

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\facebook5>cd C:\Program Files\MongoDB\Server\4.0\bin

C:\Program Files\MongoDB\Server\4.0\bin>mongo
MongoDB shell version v4.0.4
connecting to: mongodb://127.0.0.1:27017
Implicit session: session { "id" : UUID("321bfc14-beba-4f08-8ad1-bf857e2cd7e6") }
MongoDB server version: 4.0.4
Server has startup warnings:
2018-11-15T18:24:11.992+0700 I CONTROL [initandlisten]
2018-11-15T18:24:11.992+0700 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-11-15T18:24:11.992+0700 I CONTROL [initandlisten] ** Read and write access to data and configuration is u
nrestricted.
2018-11-15T18:24:11.992+0700 I CONTROL [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
```

Maka tampilannya akan seperti ini.

6. Ketikan show dbs (berfungsi untuk melihat semua database yang ada di komputer kita)

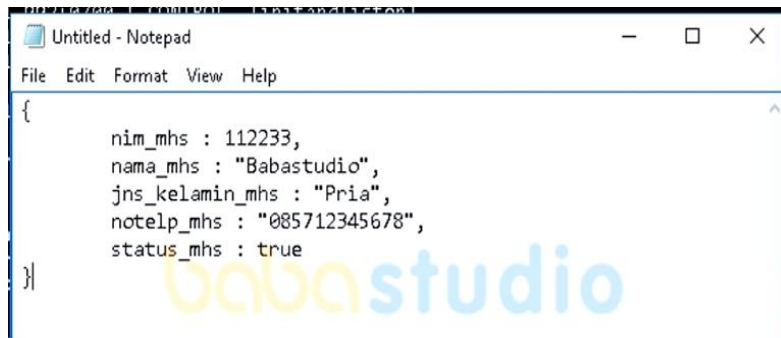
```
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
>
```

Maka akan tampil admin, config , dan local (database bawaan dari mongoDB)

7. Untuk membuat database baru ketikan “use” contoh : “use kampus”

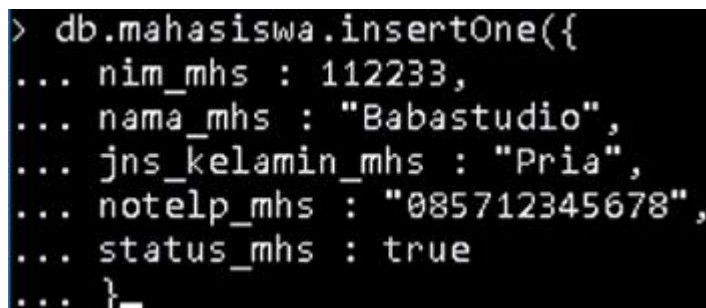
```
> use kampus
switched to db kampus
```

8. Buka notepad masing – masing lalu ketikan



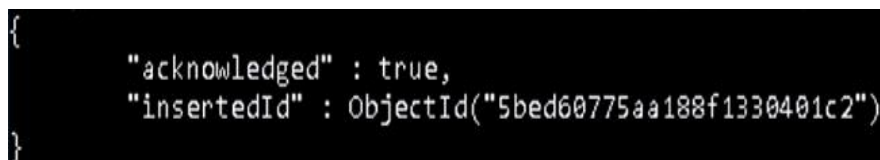
```
{
  nim_mhs : 112233,
  nama_mhs : "Babastudio",
  jns_kelamin_mhs : "Pria",
  notelp_mhs : "085712345678",
  status_mhs : true
}
```

9. Buka kembali Command Promptnya kita akan menambahkan sebuah tabel atau Collection pada mongodb yaitu dengan cara “db.nama_tabel.insertOne” contoh : “db.mahasiswa.insertOne” setelah mengetik ini coba di copy data yang sudah kita buat pada notepad sehingga menjadi seperti ini.



```
> db.mahasiswa.insertOne({
... nim_mhs : 112233,
... nama_mhs : "Babastudio",
... jns_kelamin_mhs : "Pria",
... notelp_mhs : "085712345678",
... status_mhs : true
... })
```

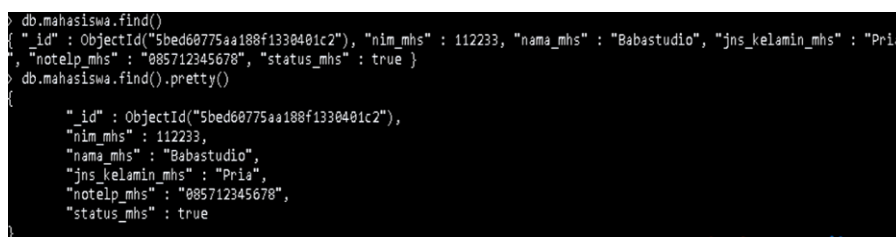
Tekan enter.



```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5bed60775aa188f1330401c2")
}
```

Kalau tampil seperti ini maka proses penyimpanan data kita telah berhasil.

10. Selanjutnya kita akan menampilkan isi tabel tersebut dengan cara “db.nama_tabel.find()” contoh : “db.mahasiswa.find()” dan apabila tampilannya kurang bagus di tambahkan “pretty()” di akhir contoh : “db.mahasiswa.find().pretty()”.



```
> db.mahasiswa.find()
{ "_id" : ObjectId("5bed60775aa188f1330401c2"), "nim_mhs" : 112233, "nama_mhs" : "Babastudio", "jns_kelamin_mhs" : "Pria", "notelp_mhs" : "085712345678", "status_mhs" : true }
> db.mahasiswa.find().pretty()
{
  "_id" : ObjectId("5bed60775aa188f1330401c2"),
  "nim_mhs" : 112233,
  "nama_mhs" : "Babastudio",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : "085712345678",
  "status_mhs" : true
}
```

11. Kita akan menambahkan beberapa data lagi ke dalam tabel mahasiswa. Untuk menambahkan tabel langsung banyak kita dapat menggunakan cara “db.nama_tabel.insertMany()” contoh : “db.mahasiswa.insertMany()”. Nah sekarang coba ketikan lagi seperti ini



```
[
{
  nim_mhs : 112233,
  nama_mhs : "Babastudio",
  jns_kelamin_mhs : "Pria",
  notelp_mhs : "085712345678",
  status_mhs : true
},
{
  nim_mhs : 112234,
  nama_mhs : "Neo",
  jns_kelamin_mhs : "Pria",
  notelp_mhs : "081234567890",
  status_mhs : true
}
]
```

Di copy lalu ketikan di command prompt seperti ini



```
> db.mahasiswa.insertMany([
... {
... nim_mhs : 112233,
... nama_mhs : "Babastudio",
... jns_kelamin_mhs : "Pria",
... notelp_mhs : "085712345678",
... status_mhs : true
... },
... {
... nim_mhs : 112234,
... nama_mhs : "Neo",
... jns_kelamin_mhs : "Pria",
... notelp_mhs : "081234567890",
... status_mhs : true
... }
... ])
```

Tekan enter.

12. Selanjutnya coba di tampilkan lagi data tersebut.

```
> db.mahasiswa.find().pretty()
{
  "_id" : ObjectId("5bed60775aa188f1330401c2"),
  "nim_mhs" : 112233,
  "nama_mhs" : "Babastudio",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : "085712345678",
  "status_mhs" : true
}
{
  "_id" : ObjectId("5bed627c5aa188f1330401c3"),
  "nim_mhs" : 112233,
  "nama_mhs" : "Babastudio",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : "085712345678",
  "status_mhs" : true
}
{
  "_id" : ObjectId("5bed627c5aa188f1330401c4"),
  "nim_mhs" : 112234,
  "nama_mhs" : "Neo",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : "081234567890",
  "status_mhs" : true
}
```

F. CARA MENGUBAH DATA PADA MONGODB

Untuk mengubah data disini kita bisa menggunakan “updateOne” atau “updateMany” untuk penulisan rumusnya seperti ini : “db.nama_tabel.updateOne({kriteria}, {\$set : { data yang di ubah } })” coba kita ketikan ini pada command prompt.

```
> db.mahasiswa.updateOne({_id : 112234},{$set: { nama_mhs : "Budi"} })
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

Untuk updateOne (mengubah data nama_mhs berdasarkan idnya 112234)

```
> db.mahasiswa.updateMany({status_mhs : true},{$set: {jns_kelamin_mhs : "wanita"} })
{ "acknowledged" : true, "matchedCount" : 4, "modifiedCount" : 4 }
```

Untuk updateMany (mengubah data jns_kelamin_mhs berdasarkan status mahasiswanya true)

G. CARA MENGHAPUS DATA PADA MONGODB

Untuk menghapus data disini bisa menggunakan “deleteOne” atau “deleteMany” untuk penulisan rumusnya seperti ini : “db.nama_tabel.deleteOne({kriteria})” coba kita ketikan ini pada command prompt.

```
> db.mahasiswa.deleteOne({nim_mhs : 112233})
{ "acknowledged" : true, "deletedCount" : 1 }
```

Untuk deleteOne (menghapus data berdasarkan nim mahasiswanya 112233)

```
> db.mahasiswa.deleteMany({status_mhs : true})
{ "acknowledged" : true, "deletedCount" : 3 }
```

Untuk deleteMany (menghapus data berdasarkan status mahasiswa true)

H. CARA MENAMPIL DATA BERDASARKAN KRITERIA TERTENTU PADA MONGODB

Untuk menampilkan data berdasarkan kriteria tertentu disini kita bisa menggunakan rumus ini : “db.nama_tabel.find({kriteria})” , coba kita ketikan ini pada command prompt.

```
> db.mahasiswa.find({nim_mhs : 112233}).pretty()
{
  "_id" : ObjectId("5bee435a253c9ba7e67eb3ba"),
  "nim_mhs" : 112233,
  "nama_mhs" : "Babastudio",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : 8123456789,
  "umur_mhs" : 20,
  "status_mhs" : true
}
```

Untuk menampilkan data berdasarkan kriteria nim_mhs : 112233

Kriteria bisa juga seperti :

Nama	Simbol	MongoDB
Tidak sama dengan	!=	\$ne
Lebih dari	>	\$gt
Lebih dari atau sama dengan	>=	\$gte
Kurang dari	<	\$lt
Kurang dari atau sama dengan	<=	\$lte

Contohnya :

```
> db.mahasiswa.find({nim_mhs : {$ne : 112233}}).pretty()
{
  "_id" : ObjectId("5bee435a253c9ba7e67eb3bb"),
  "nim_mhs" : 112234,
  "nama_mhs" : "Neo",
  "jns_kelamin_mhs" : "Pria",
  "notelp_mhs" : 8123456789,
  "umur_mhs" : 20,
  "status_mhs" : true
}
```

BAB II

A. TIPE – TIPE DATA PADA MONGODB

Mongodb mempunyai beberapa tipe data dalam penyimpanannya. Tipe – tipe datanya sebagai berikut :

1. String

Tipe data string adalah tipe data yang digunakan untuk menyimpan sebuah kata – kata pada database. Contohnya : { nama : “Babastudio Keren” }

```
> db.murid.insertOne({ nama : "Babastudio Keren"})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("5beffd2be58961820a3f06bb")
}
```

2. Int

Tipe data int atau integer adalah tipe data yang digunakan untuk menyimpan sebuah angka pada database. Disini untuk integer di bagi ke menjadi 2 yaitu int dan number long, perbedaannya adalah dari segi banyaknya penyimpanan untuk int bisa menampung data kurang lebih “2.000.000.000” sedangkan untuk number long bisa menampung data kurang lebih “9.000.000.000.000.000.000”. Contohnya : { harga : 1000000 }. *(untuk angka tidak menggunakan kutip 2, yang menggunakan kutip 2 hanya untuk tipe data string & untuk tipe data ini jangan menulis menggunakan titik ya).*

3. Decimal

Tipe data decimal adalah tipe data yang digunakan untuk menyimpan sebuah angka pecahan pada database. Contohnya : { nilai : 50.5 }. *(untuk koma disini menggunakan titik ya tidak menggunakan koma).*

```
> db.murid.updateOne({}, { $set : { nilai : 50.5 } })
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

4. Boolean

Tipe data boolean adalah tipe data yang hanya mengenal “true” dan “false” atau benar dan salah saja. Contohnya { status_aktif : true } *(untuk penulisan true atau false disini juga tidak memakai kutip 2 ya)*

```
> db.murid.updateOne({}, { $set : { status_aktif : true } })
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

5. Date

Tipe data date adalah tipe data yang digunakan untuk menyimpan sebuah data tanggal pada database. Contohnya { tanggal_masuk : ISO("2018-11-20") } untuk tanggal yang sudah ditetapkan atau kalau ingin menyimpan tanggal sekarang bisa menggunakan new date() contohnya { hari_ini : new Date() }.

```
> db.murid.updateOne({}, { $set : { tanggal_masuk : ISODate("2018-09-09") } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.murid.findOne()
{
  "_id" : ObjectId("5beffd2be58961820a3f06bb"),
  "nama" : "Babastudio Keren",
  "status_aktif" : true,
  "nilai" : 50.5,
  "tanggal_masuk" : ISODate("2018-09-09T00:00:00Z")
}
> db.murid.updateOne({}, { $set : { hari_ini : new Date() } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

6. Array

Tipe data array adalah tipe data yang digunakan untuk menyimpan beberapa data dalam 1 kolom pada database. Contohnya { hobi : ["sepak bola","renang"] }

7. Object

Tipe data object adalah tipe data yang digunakan untuk menyimpan beberapa data dalam 1 kolom pada database (hampir sama dengan array tetapi kalau array data yang ada di dalamnya kita bisa menyebutnya hobi[0] untuk baris pertama nah sedangkan untuk object ini kita bisa memberikan nama data pada setiap barisnya). Contohnya

```
{
  status : {
    status_aktif : true,
    status_lulus : false
  }
}
```

8. ObjectID

Tipe data objectid adalah tipe data yang digunakan untuk menyimpan id pada database. Contohnya

```
{ _id : ObjectId("4beffd2be5896hror92k123a") }
```

```
"_id" : ObjectId("5beffd2be58961820a3f06bb")
```

B. RELASI

Relasi adalah suatu hubungan antar tabel pada database dimana suatu tabel mempunyai data yang sama dengan data yang ada di tabel lainnya. Terdapat 3 macam relasi yaitu :

1. Relasi one to one

Hubungan antar tabel dimana tabel A adalah data master dan tabel B harus mempunyai data yang ada pada tabel A. Contohnya :

tabel pasien

nama : "Babastudio"
umur : 20
penyakit : sakit_0001

tabel penyakit

_id : sakit_0001
penyakit : flu

Disini tabel penyakit merupakan data master atau data inti ya dan tabel pasien disini mempunyai data yang ada pada tabel penyakit.

Coba di buat sebuah database bernama rumah_sakit, dan masukkan data pasien dan penyakit.

```
> var penyakit_id = db.pasien.findOne().penyakit
```

Buat sebuah variabel baru (variabel berfungsi untuk penampungan data sementara sedangkan "db.pasien.findOne().penyakit" berfungsi untuk menampilkan data sakit_0001 jadi variabel penyakit_id berisikan sakit_0001.

Lalu ketikan seperti ini :

```
> db.penakit.findOne({_id : penyakit_id})  
{ "_id" : "sakit_0001", "penyakit" : "flu" }
```

Disini dengan menyimpan data penyakit pasien kita dapat mencari nama penyakitnya di tabel penyakit.

2. Relasi one to many

Sebuah hubungan antara tabel dimana tabel A memiliki sebuah data yang bisa di pakai pada tabel B data tersebut bisa banyak data atau beberapa data saja. Contohnya :

```
tabel pelanggan
```

```
_id : "PL0001",  
nama_pelanggan : "Babastudio"
```

```
tabel transaksi
```

```
_id : "TR0001",  
tanggal_transaksi : New Date(),  
total_harga : 100000  
id_pelanggan : "PL0001"
```

tabel pelanggan adalah tabel masternya sedangkan tabel transaksi adalah yang akan di relasikan.

Coba di buat sebuah database bernama transaksi_nama-masing-masing, dan masukkan data pelanggan dan transaksi (tambahkan data transaksi minimal 3 data, untuk isi datanya bebas asalkan sesuai dengan format data di atas kecuali id_pelanggan diisi sama dengan di atas).

Kalau sudah coba ketika “db.transaksi.find().pretty()” nah maka akan muncul semua datanya. Disini untuk id transaksinya berbeda – beda tetapi id_pelanggannya tetap sama nah itu relasi dari one to many atau satu ke banyak ya (dari 1 data pelanggan bisa menghasilkan banyak data pada transaksi).

3. Relasi many to many

Sebuah hubungan antara tabel dimana tabelnya ini ada banyak. Hubungannya itu bisa banyak tabel misalkan tabel A dengan tabel B dan tabel C dengan tabel B seperti itu. Contohnya :

```
tabel detail transaksi  
{  
no_trans : "TR0001",  
barang : "Gelas",  
jumlah : 3  
}
```

Coba kita buat tabel detail transaksi pada tabel transaksi_nama-masing-masing yang sudah di buat sebelumnya. Selanjutnya coba di tambahkan datanya lagi selain gelas minimal 3 data dan untuk no_trans nya jangan di rubah atau disamakan saja dengan data di atas.

Disini tabel transaksi yang sebelumnya sudah berrelasi dengan tabel pelanggan bisa berrelasi lagi dengan tabel yang berbeda yaitu tabel detail transaksi itu relasi dari One to Many.

C. LOOKUP

Lookup adalah sebuah metode yang digunakan untuk menampilkan data seperti find tetapi di lookup ini kita dapat menampilkan beberapa tabel untuk di tampilkan (lookup ini bisa berjalan kalau tabelnya sudah berrelasi). Contohnya :

```
db.collection.aggregate({
  $lookup : {
    from : collection2,
    localField : "_id",
    foreignField : "_id",
    as : "join"
  }
})
```

Keterangan :

Aggregate : untuk mengelompokan data

Lookup : untuk mengabungkan data

From : diisi tabel ke 2 yang akan di tampilkan

localField : diisi dengan nama data yang ada di tabel pertama (data ini harus data yang nanti sama dengan data yang ada di tabel ke 2)

foreignField : sama dengan localField tetapi untuk foreignField diisi dengan nama data yang ada di tabel kedua

as : as atau alias dapat diisi bebas karena disini jika nanti data kita mau di tampilkan kita cukup memanggil nama yang sudah di aliaskan saja.

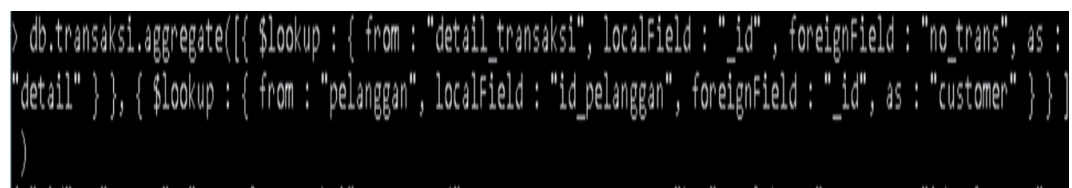
```
> db.transaksi.aggregate({ $lookup : { from : "detail_transaksi", localField : "_id", foreignField : "no_trans", as : "detail" } } )
{ "_id" : "TR0001", "tanggal_transaksi" : ISODate("2018-11-21T13:29:04.413Z"), "total_harga" : 100000, "id_pelanggan" : "PL0001", "detail" : [ { "_id" : ObjectId("5bf7fb6d80c75c064d3e09ee"), "no_trans" : "TR0001", "barang" : "Gelas", "jumlah" : 3 }, { "_id" : ObjectId("5bf7fc1180c75c064d3e09ef"), "no_trans" : "TR0001", "barang" : "Piring", "jumlah" : 12 } ] }
{ "_id" : "TR0002", "tanggal_transaksi" : ISODate("2018-11-21T13:29:26.507Z"), "total_harga" : 100000, "id_pelanggan" : "PL0001", "detail" : [ ] }
{ "_id" : "TR0003", "tanggal_transaksi" : ISODate("2018-11-21T13:29:32.855Z"), "total_harga" : 100000, "id_pelanggan" : "PL0001", "detail" : [ ] }
```

Untuk menampilkan 3 tabel atau lebih sekaligus kita dapat menggunakan cara ini:

```
db.collection.aggregate(  
  [  
    {  
      $lookup : {  
        from : collection2,  
        localField : "_id",  
        foreignField : "_id",  
        as : "join"  
      }  
    },  
    {  
      $lookup : {  
        from : collection3,  
        localField : "_id",  
        foreignField : "_id",  
        as : "join"  
      }  
    }  
  ]  
)
```

Keterangan :

Pembedanya adalah pada kurung [] nya ya



```
> db.transaksi.aggregate([ { $lookup : { from : "detail_transaksi", localField : "_id", foreignField : "no_trans", as : "detail" } }, { $lookup : { from : "pelanggan", localField : "id_pelanggan", foreignField : "_id", as : "customer" } } ] )
```

D. VALIDASI

Validasi adalah sebuah metode untuk pengecekan suatu data yang di masukkan, misalkan pada saat kita login pada suatu website kita ketikan asal – asalan maka akan muncul peringatan. Coba buat database terlebih dahulu dengan nama “barang_nama-masing-masing”. Nah sekarang coba kita ketikan seperti ini pada notepad ++ atau notepad biasa.

```

db.createCollection("barang", {
  validator : {
    $jsonSchema : {
    }
  }
})

```

Lalu kita masukan ini pada json schema

```

$jsonSchema : {
  bsonType : "object",
  required : ["namabarang", "hargabarang", "jenisbarang", "description"],
  properties : {
  }
}

```

Selanjutnya masukan propertiisnya

```

namabarang : {
  bsonType : "string",
  description : "Nama Barang harus berupa karakter dan tidak boleh kosong!"
},
hargabarang : {
  bsonType : "number",
  description : "Harga Barang harus berupa angka dan tidak boleh kosong!"
},
jenisbarang : {
  bsonType : "objectId",
  description : "Jenis Barang harus berupa ID dan tidak boleh kosong!"
},
description : {
  bsonType : "array",
  description : "Description harus berupa array dan tidak boleh kosong!"
}

```

Pada description karna di berupa array coba di tambahkan lagi seperti ini

```

description : {
  bsonType : "array",
  description : "Description harus berupa array dan tidak boleh kosong!",
  items : {
    merk : {
      bsonType : "string",
      description : "Merk harus berupa karakter dan tidak boleh kosong!"
    },
    ukuran : {
      bsonType : "string",
      description : "Ukuran harus berupa karakter dan tidak boleh kosong!"
    },
    stok : {
      bsonType : "number",
      description : "Stok harus berupa angka dan tidak boleh kosong!"
    }
  }
}

```

Lalu tambahkan script ini di bawah items

```
items : {  
  bsonType : "object",  
  required : ["merk", "ukuran", "stok"],  
  properties : {  
    merk : {  
      bsonType : "string",  
      description : "Merk harus beru"  
    },  
  },  
}
```

Jangan lupa di kasih) di bagian paling bawah untuk menutup createCollection.

Keterangan :

createCollection : untuk membuat tabel

validator : untuk membuat validasi pada json

\$jsonSchema : untuk masuk ke dalam fungsi json

bsonType : tipe data

required : data apa saja yang pada saat di tambahkan datanya itu harus diisi

properties : peraturan – perturan yang akan di buat untuk validasi data tersebut

items : untuk data bertipe array

Sekarang coba di jalankan. Dan apabila kita inputkan asal – asalan maka akan tampil seperti ini :

```
2018-11-24T17:51:35.681+0700 E QUERY [js] WriteError: Document failed validation :  
WriteError({  
  "index" : 0,  
  "code" : 121,  
  "errmsg" : "Document failed validation",  
  "op" : {  
    "_id" : ObjectId("5bf92d37ba95b118fa6c4dd6"),  
    "namabarang" : "Baju Gamis",  
    "hargabarang" : 200000,  
    "jenisbarang" : "thth",  
    "description" : [  
      {  
        "merk" : "Syari",  
        "ukuran" : "All size",  
        "stok" : 10  
      }  
    ]  
  }  
})  
WriteError@src/mongo/shell/bulk_api.js:461:48  
Bulk/mergeBatchResults@src/mongo/shell/bulk_api.js:841:49  
Bulk/executeBatch@src/mongo/shell/bulk_api.js:906:13  
Bulk/this.execute@src/mongo/shell/bulk_api.js:1150:21  
DBCollection.prototype.insertOne@src/mongo/shell/crud_api.js:252:9  
@(shell):1:1
```

BAB III

A. MongoDB Help

Mongodb help adalah suatu script atau sintak yang digunakan untuk mengetahui fitur – fitur atau script yang ada pada mongodb. Untuk mengetahui script apa saja di mongodb disini ada bagian – bagiannya yaitu :

1. mongod help

Digunakan untuk menampilkan script – script apa saja yang bisa kita jalankan pada fungsi mongod. Penulisannya seperti ini

```
C:\Program Files\MongoDB\Server\4.0\bin>mongod --help
```

2. mongo –help

Digunakan untuk menampilkan script – script apa saja yang bisa kita jalankan pada fungsi mongo. Penulisannya seperti ini

```
C:\Program Files\MongoDB\Server\4.0\bin>mongo --help
```

3. help (jika sudah masuk pada mongo)

Digunakan untuk menampilkan script – script apa saja yang bisa kita jalankan pada fungsi sesudah mongo dijalankan. Sebelumnya untuk masuk ke fungsi ini jalankan terlebih dahulu mongo pada command prompt lalu ketika sudah barulah kita bias menggunakan fungsi ini. Penulisannya seperti ini

```
> help
```

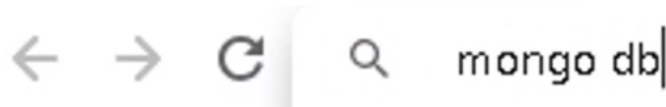
4. db.help()

Digunakan untuk menampilkan script – script apa saja yang bisa kita jalankan pada fungsi db. Script ini bisa dijalankan ketika kita sudah menggunakan suatu database, Penulisannya seperti ini

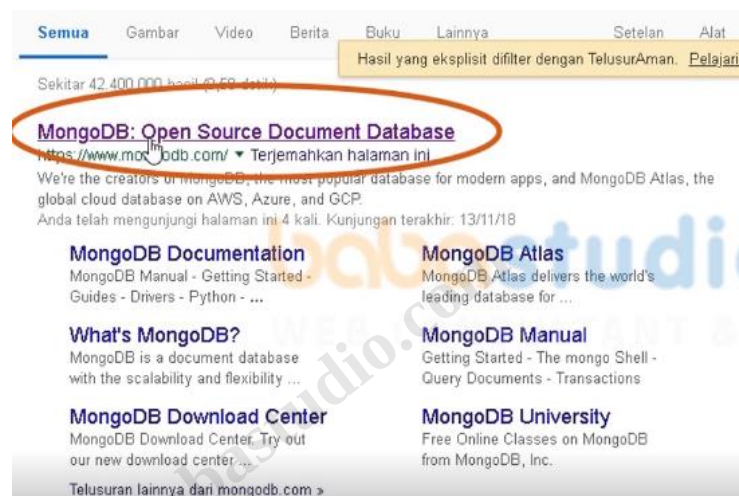
```
> use kampus  
switched to db kampus  
> db.help()
```

B. MongoDB Compass

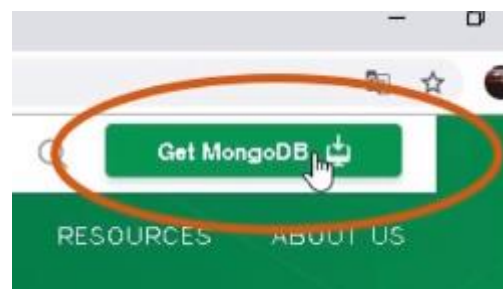
MongoDB compass adalah sebuah software atau GUI (*graphical user interface*) pada mongodb yang digunakan untuk memproses data – data yang ada pada mongodb. Dengan menggunakan software ini kita bisa tidak menggunakan command prompt lagi untuk menjalankan sintak – sintak seperti buat database, buat tabel , menambahkan data , dll. Untuk mendownload software ini coba kita buka browser masing – masing.



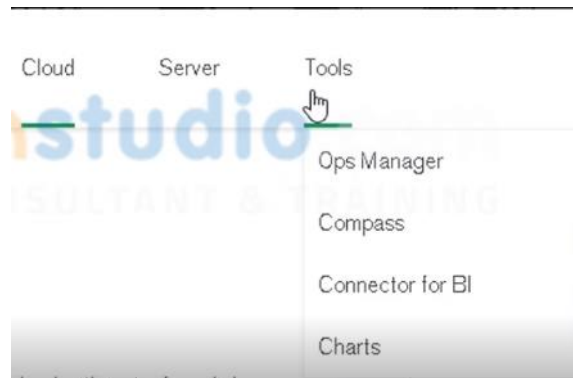
Ketikan mongo db pada browser masing – masing



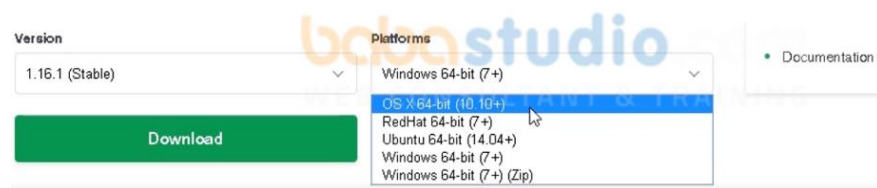
Lalu pilih yang seperti ini



Klik get MongoDB



Selanjutnya kita pilih tool yang berada di tengah bawah lalu kita pilih compass.



Silahkan pilih platforms sesuai dengan computer / laptop masing – masing lalu tekan download.

Download Now

Business email

First Name

Last Name

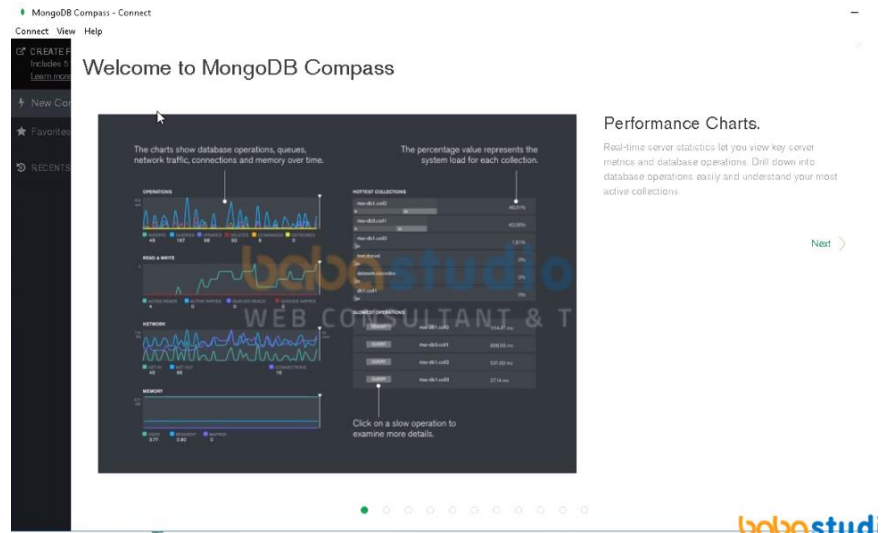
Business phone

Company

IT Executive (CIO, CTO, VP Engineering, etc) ✓

☒ Check here to indicate that you have read and agree to the terms of the [Customer Agreement](#).

Maka disini sebelum mendownload disini kita diminta untuk mengisi sebuah form terlebih dahulu. silahkan diisi lalu klik submit. Maka mongodb compass akan terdownload otomatis. Kalau sudah selesai di download silahkan buka foldernya lalu cari file mongodb-compass coba di klik 2x.



Kalua sudah selesai maka tampilannya akan seperti ini. Lalu di klik next – next saja dan kalau sudah tekan get started.

Privacy Settings

To enhance the user experience, Compass can integrate with 3rd party services, which requires external network requests. Please choose from the settings below:

- ☒ **Enable Product Feedback Tool**
Enables a tool for sending feedback or talking to our Product and Development teams directly from Compass.
- ☒ **Enable Geographic Visualizations**
Allow Compass to make requests to a 3rd party mapping service.
- ☒ **Enable Crash Reports**
Allow Compass to send crash reports containing stack traces and unhandled exceptions.
- ☒ **Enable Usage Statistics**
Allow Compass to send anonymous usage statistics.
- ☒ **Enable Automatic Updates**
Allow Compass to periodically check for new updates.

With any of these options, none of your personal information or stored data will be submitted.
[Learn more: MongoDB Privacy Policy](#)

Start Using Compass

Klik start using compass. Lalu tekan Connect, maka akan tampil database – database yang telah dibuat sebelumnya. Sekarang kita coba untuk membuat sebuah database baru dengan cara



Klik create database

A screenshot of the 'Create Database' form. It has two input fields: 'Database Name' with the value 'buku_babastudio' and 'Collection Name' with the value 'buku'. Below these are two checkboxes: 'Capped Collection' and 'Use Custom Collation', both of which are unchecked. A light blue informational box states: 'Before MongoDB can save your new database, a collection name must also be specified at the time of creation. [More Information](#)'. At the bottom right, there are two buttons: 'CANCEL' and 'CREATE DATABASE', with a mouse cursor clicking on the latter.

Kita buat database baru bernama buku_nama-masing-masing dengan collection buku lalu klik create database. Untuk menambahkan data coba kita klik database yang sudah kita buat, klik lagi collection/tabel yang sudah di buat.



Lalu klik insert document

A screenshot of the 'Insert Document' form. It shows a JSON document structure with fields: '_id' (value: 'buku0001'), 'nama_buku' (value: 'Buku babastudio'), 'deskripsi' (value: 'Object'), 'penerbit' (value: 'Babastudio'), and 'tahun_terbit' (value: '2018'). To the right of the document, there are data type indicators: 'String' for _id, 'String' for nama_buku, 'Object' for deskripsi, 'String' for penerbit, and 'String' for tahun_terbit. At the bottom right, there are 'CANCEL' and 'INSERT' buttons, with a mouse cursor hovering over the 'INSERT' button. There are also blue circles with numbers 1, 2, and 3 pointing to the 'tahun_terbit' field, the 'nama_buku' field, and the data type indicators respectively.

Isi seperti ini.

Keterangan :

1. Nomor satu adalah nama kolom atau nama data
2. Nomor dua adalah isi dari suatu kolom
3. Nomor tiga adalah tipe data dari data yang diisi

Selanjutnya coba kita tekan tombol insert.

```
> { "_id": "buku0001",  
  "nama_buku": "Buku babastudio"  
> deskripsi: Object
```

Maka datanya akan tersimpan & akan langsung di tampilkan seperti ini.



Pada mongoddb compass juga terdapat Edit Document, Copy Document, Clone Document, dan Delete Document.

C. Ordered

Ordered adalah sebuah validasi mongoddb yang memberi pilihan kepada pengguna apabila script yang di ketika error mau tetap di teruskan atau tidak di teruskan. Coba kita buat sebuah database baru dengan nama hobi, lalu kita isi datanya seperti berikut:

```
> use hobi  
switched to db hobi  
> db.hobi.insertMany([{"_id": "sepakbola", "nama": "Sepak Bola"}, {"_id": "renang", "nama": "Renang"}])  
{ "acknowledged": true, "insertedIds": [ "sepakbola", "renang" ] }
```

Jika di tampilkan maka akan seperti ini

```
> db.hobi.find()  
{ "_id": "sepakbola", "nama": "Sepak Bola" }  
{ "_id": "renang", "nama": "Renang" }
```

Sekarang kita coba menambahkan data dengan orderednya

```
db.hobi.insertMany(  
  
  [  
  
    {
```

```

        _id : "Pencak Silat",
        Nama : "pencak silat"
    },
    {
        _id : "Renang",
        Nama : "Renang"
    },
    {
        _id : "Judo",
        Nama : "Judo"
    },
    {
        ordered:true
    }
]
)

```

Keterangan:

Disini ordered bertipe Boolean atau hanya bisa mengenal true dan false saja. jika dijalankan tanpa ordered maka akan error tetapi apabila kode diatas di jalankan maka akan muncul error juga tetapi jika errornya itu mengarah ke data yang kedua berarti data yang pertama akan tersimpan dan data yang kedua dan seterusnya tidak bisa tersimpan. Dapat disimpulkan jika orderednya true maka akan membaca data sampai terjadi error sedangkan kalau orderednya false akan membaca semua data kecuali data yang sudah ada atau data error.

D. Export Database

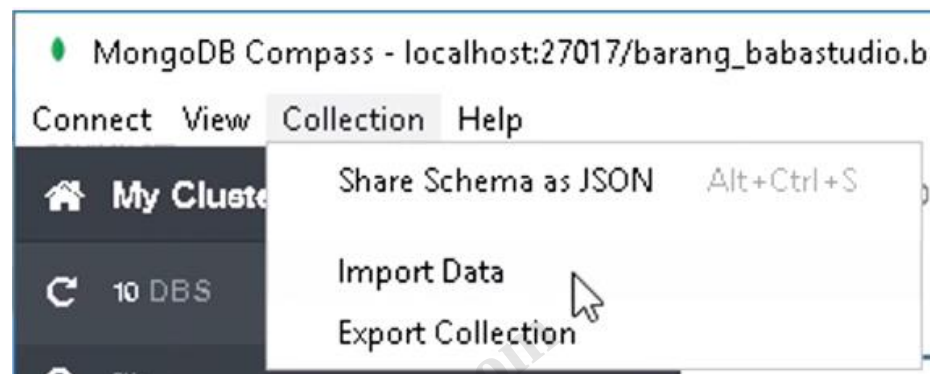
Export database adalah proses membuat data cadangan dengan cara menyalin atau membuat arsip data komputer sehingga data tersebut dapat digunakan kembali apabila terjadi kerusakan atau kehilangan. Export data di mongodb ada 2 cara yaitu dengan file : .json dan .csv (berformat excel)

1. Export data menggunakan file json
Sebelum kita menexport suatu data coba kita buat file json nya terlebih dahulu dengan cara :

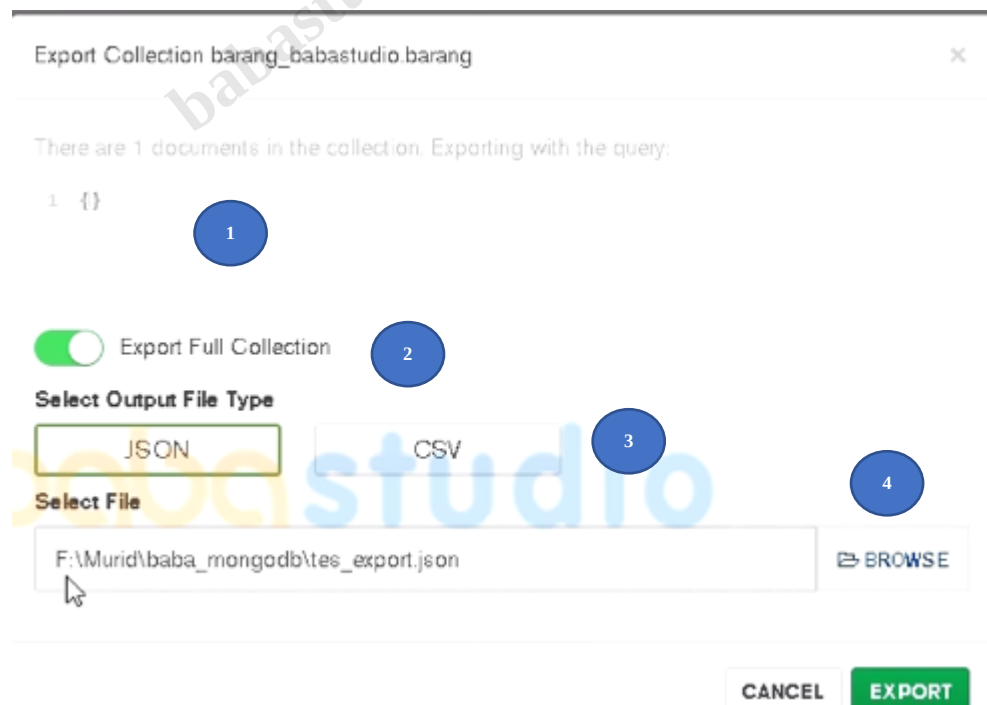
- Buka notepad
- Lalu kita save as
- Tentukan direktorinya terlebih dahulu
- Ubah save as type nya menjadi All Files
- Ubah namanya menjadi tes_export.json
- Lalu tekan save

A. Pengimplementasiannya pada mongodb Compass

Sekarang coba kita buka mongodb compass dan buka database & tabel yang akan kita export.



Kalau sudah di buka coba kita klik collection di menu kiri sebelah atas lalu kita klik Export Collection. Maka akan muncul seperti ini :



Kemudian klik export full collection lalu browse file json yang telah di buat sebelumnya lalu tekan export.

Keterangan :

1. Untuk nomor 1 apabila kita ingin mengetik sendiri script – script yang kita ingin backup kita bisa tulis disini.
2. Untuk nomor 2 apabila kita tidak mau mengetik kita cukup memilih ini maka semua scriptnya pada tabel ini akan tersimpan
3. Untuk nomor 3 ini adalah file yang bisa di export pada monggodb yaitu json dan csv
4. Untuk nomor 4 disini kita browser untuk memilih file json yang telah kita buat sebelumnya.

B. Pengimplementasiannya pada command prompt

Buatkan terlebih dahulu file json dan simpan file tersebut didalam direktori tempat penginstalan monggodb atau pada folder bin lalu coba kita buka commad prompt. Lalu coba commad promptnya lalu pada command prompt masuk ke direktori monggodb berada lalu coba ketikan seperti ini :

```
mongoexport --db barang_babastudio --collection barang --out barang.json
```

Keterangan :

- Mongoexport adalah rumus untuk mengexport sebuah data
- Db adalah nama database yang akan diexport
- Colletion adalah nama tabel yang akan di export
- Out merupakan keluaran dari yang kita export
- Kode ini harus di jalankan dalam user administrator pada komputer agar kita bisa mengedit datanya kalau tidak maka akan terjadi error (khusus yang belajar pada laptop ataupun computer sendiri)

2. Export data menggunakan file csv

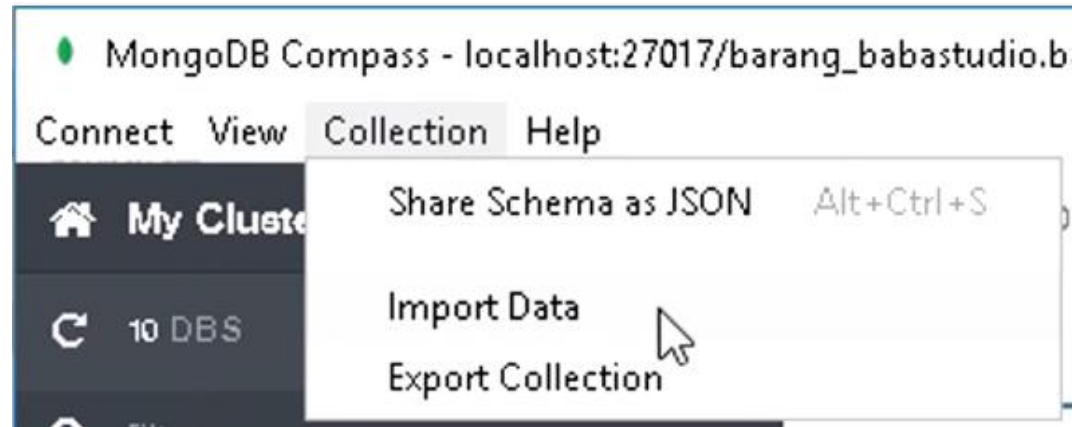
Untuk mengexport data dengan menggunakan file csv hamper sama dengan export data pada json yaitu coba kita buat file csvnya terlebih dahulu dengan cara :

- Buka Microsoft Excel
- Buat new file
- Klik save as
- Tentukan direktorinya terlebih dahulu
- Ubah save as type nya menjadi All Files
- Ubah namanya menjadi tes_export.csv
- Lalu tekan save

A. Pengimplementasikannya pada mongodb Compass

Sekarang coba kita buka mongodb compass dan buka database & tabel yang akan kita export.

Sekarang coba kita buka mongodb compass dan buka database & tabel yang akan kita export.



Kalau sudah di buka coba kita klik collection di menu kiri sebelah atas lalu kita klik Export Collection. Pada select output file type sekarang kita ganti dengan csv. Lalu kita browse pilih data csv yang sudah di buat lalu klik export.

B. Pengimplementasikannya pada command prompt

Buatkan terlebih dahulu file csv dan simpan file tersebut didalam direktori tempat penginstalan mongodb atau pada folder bin lalu coba kita buka commad prompt. Lalu coba commad promptnya lalu pada command prompt masuk ke direktori mongodb berada lalu coba k

```
mongoexport --db barang_babastudio --collection barang --type=csv -fields  
_id,namabarang,hargabarang,jenisbarang,description --out barang.csv
```

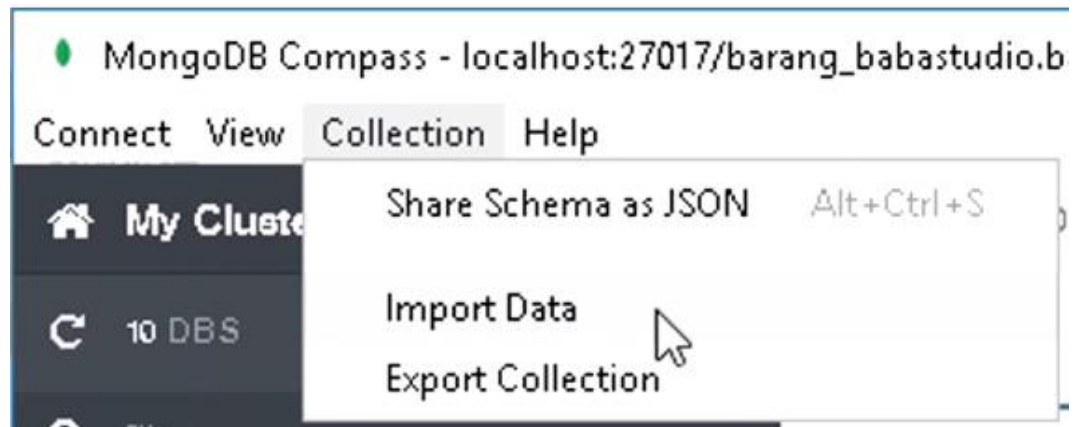
k
an seperti ini :

Keterangan :

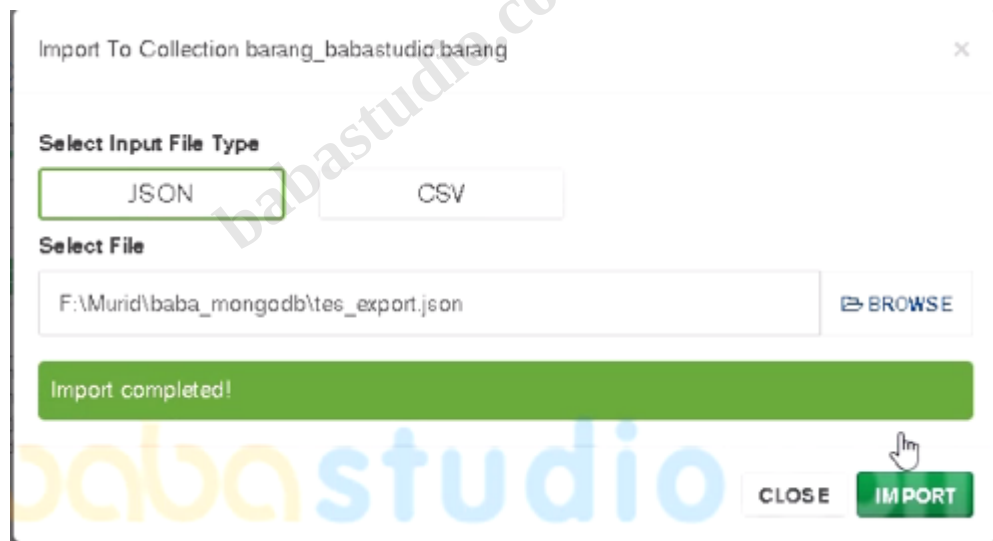
- Type adalah tipe dari file yang nanti kita export
- Fields adalah nama kolom/data pada database

E. Import Database

Import database adalah cara untuk memasukkan data yang sebelumnya telah di export menjadi data csv atau json. Untuk import database coba kita buka aplikasi mongodb compass dan buka database & tabel yang akan kita import.



Kalau sudah di buka coba kita klik collection di menu kiri sebelah atas lalu kita klik Import Collection. Maka akan muncul seperti ini :



Lalu coba browse file yang sudah di export dalam bentuk json terlebih dahulu, selanjutnya tekan import. Maka datanya akan masuk ke dalam database kita (langkah import dapat dilakukan jika data pada tabel yang sebelumnya di export sudah terhapus/tabel tersebut sekarang berada tidak ada datanya).

S

e

```
mongoimport --db barang_babastudio --collection barang --file  
f:\Murid\baba_mongodb\tes_export.json
```

rang jika mengimport data pada command prompt cara seperti ini :

Keterangan :

- Mongoimport digunakan sebagai rumus untuk memasukan data pada database yang berupa json ataupun csv
- File digunakan untuk menyebutkan direktori file json atau csv yang akan di import

babastudio.com

BAB IV

A. Konsep Method, Filter, Operator

Untuk penulisan rumus pada mongodb kita membutuhkan ketiga script ini yaitu Method, Filter, dan Operator. Method adalah metode yang di pakai sebagai aksi yang akan digunakan misalkan `find()`, `insertOne()`, `deleteOne()`, dll. Untuk filter adalah sebuah fungsi untuk menyaring sebuah data agar datanya tersebut bisa di kelompokkan dan tidak tampil semua pada saat kita mau menampilkannya contoh `find({ umur : 20 })`. Sedangkan untuk operator adalah sebuah fungsi yang digunakan agar menyaring data pada filter bisa lebih spesifik lagi misalkan seperti contoh pada filter disitu hanya menyaring data umur yang 20 tahun saja sedangkan ketika kita memkasi operator disini bisa menyaring berdasarkan umur yang lebih dari 20 atau di bawah 20 tahun dan dll. Contohnya :

```
db.mahasiswa.find({ umur : { $gt : 20 } })
```

-  = Method
-  = Filter
-  = Operator

B. \$in & \$nin

\$in adalah sebuah operator yang berfungsi hampir menyerupai sama dengan (=) tetapi kalau dalam operator ini kita bisa menampilkan data lebih dari 1 data. Rumusnya adalah seperti ini :

```
db.tabel.find({ data : { $in : [ nama1,nama2 ] } }).pretty()
```

Contohnya pada command prompt :

```
> use hobi
switched to db hobi
> db.hobi.find().pretty()
{ "_id" : "sepakbola", "nama" : "Sepak Bola" }
{ "_id" : "renang", "nama" : "Renang" }
{ "_id" : "pencaksilat", "nama" : "Pencak Silat" }
{ "_id" : "mancing", "nama" : "Mancing" }
{ "_id" : "judo", "nama" : "Judo" }
{ "_id" : "basket", "nama" : "basket" }
> db.hobi.find({ nama : { $in : ["basket","Judo"] } }).pretty()
{ "_id" : "judo", "nama" : "Judo" }
{ "_id" : "basket", "nama" : "basket" }
```

Sedangkan untuk \$nin adalah kebalikan dari \$in yang berfungsi untuk menampilkan data yang tidak di tampilkan pada saat di \$in di jalankan. Rumusnya adalah seperti ini :

```
db.tabel.find({ data : { $in : [ nama1,nama2 ] } } ).pretty()
```

Contohnya pada command prompt :

```
> db.hobi.find({ nama : { $nin : ["basket","Judo"] } } ).pretty()
{ "_id" : "sepakbola", "nama" : "Sepak Bola" }
{ "_id" : "renang", "nama" : "Renang" }
{ "_id" : "pencaksilat", "nama" : "Pencak Silat" }
{ "_id" : "mancing", "nama" : "Mancing" }
```

C. \$and, \$or, \$not

And, or , not adalah sebuah operator yang digunakan untuk menyaring data dengan syarat tertentu agar datanya bisa ditampilkan. \$and adalah sebuah operator yang digunakan untuk menyaring data dengan 2 atau lebih kriteria dimana semua kriterianya itu harus benar untuk menampilkan datanya. Ketentuan dari \$and adalah seperti ini :

Kriteria 1	Kriteria 2	Hasil
Berhasil	Gagal	Gagal
Gagal	Berhasil	Gagal
Gagal	Gagal	Gagal
Berhasil	Berhasil	Berhasil

Jika kita simpulkan apabila data yang kita saring atau sortir berhasil melewati salah satu kriteria maka hasilnya akan tetap gagal karena pada operator \$and datanya akan berhasil apabila telah berhasil melewati ke dua kriteria tersebut. Rumusnya adalah sebagai berikut :

```
db.tabel.find( { $and : [ { data1 : isi1 }, { data2 : isi2 } ] } ).pretty()
```

\$or adalah sebuah operator yang digunakan untuk menyaring data dengan 2 atau lebih kriteria dimana jika salah satu kriteria terpenuhi maka akan menampilkan datanya. Ketentuan dari \$or adalah seperti ini :

Kriteria 1	Kriteria 2	Hasil
Berhasil	Gagal	Berhasil
Gagal	Berhasil	Berhasil
Gagal	Gagal	Gagal

Berhasil	Berhasil	Berhasil
----------	----------	----------

Jika kita simpulkan apabila data yang kita saring atau sortir berhasil melewati salah satu kriteria maka hasilnya akan berhasil. Rumusnya adalah sebagai berikut :

```
db.tabel.find( { $or : [ { data1 : isi1 }, { data2 : isi2 } ] } ).pretty()
```

\$not adalah sebuah operator yang digunakan untuk menyaring data dengan menampilkan data yang tidak sama dengan data yang dimasukan. Rumusnya adalah sebagai berikut :

```
db.tabel.find( { data : { $not : { $eq : isi } } } ).pretty()
```

Keterangan :

- \$eq adalah equal atau sama dengan(=)
- \$ne adalah not equal atau tidak sama dengan(!=)
- \$eq atau \$ne rumus yang harus digunakan dalam menggunakan \$not kalau tidak di gunakan salah satu rumus ini maka akan error.

D. \$exists, \$type, \$regex

1. \$exists

Operator exists adalah operator yang digunakan untuk pengecekan data apakah data di suatu tabel ada atau tidak. Contohnya pada suatu tabel barang terdapat data gelas dan piring. Pada data gelas terdapat sebuah data diskon tetapi pada data piring tidak ada data diskon nah pada kasus ini kita dapat memakai \$exists untuk mengecek keberadaan suatu datanya. Rumusnya adalah sebagai berikut :

```
db.tabel.find( { data : { $exists : true } } ).pretty()
```

2. \$type

Operator type adalah operator yang digunakan untuk menyaring atau mensortir data berdasarkan suatu tipe data pada sebuah data tertentu. Contohnya pada tabel barang terdapat price yang bertipe data number lalu kita akan mencarinya berdasarkan type data number maka akan muncul data barang apabila kita mencarinya berdasarkan type data string maka hasilnya tidak akan muncul. Rumusnya adalah sebagai berikut :

```
db.tabel.find( {data : { $type : "tipe-datanya" } } ).pretty()
```

3. \$regex

Operator regex adalah operator yang digunakan untuk mencari sebuah data berdasarkan kata atau huruf tertentu. Misalkan kita mempunyai data barang dengan nama "Gelas Babastudio" lalu kita mau mencari data yang Namanya babastudio maka kita bisa menggunakan operator ini. Contoh penulisan operator regex :

```
db.tabel.find( {data : { $regex : /isi-yang-mau-dicari/ } } ).pretty()
```

Keterangan :

- isi-yang-mau-dicari bisa diisi dengan nama atau huruf yang dicari misalkan mau mencari babastudio kita bisa menulis /babastudio/. Selain // kita juga dapat menggunakan :

/^ isi/ = untuk pencarian nama sesuai huruf belakangnya misalkan pada pencarian di atas kita menuliskannya /^studio/
/isi\$/ = untuk pencarian nama sesuai huruf depannya misalkan pada pencarian di atas kita menuliskannya /baba\$/

E. \$expr

Operator expr adalah sebuah operator yang berfungsi untuk membandingkan suatu data tertentu. Misalkan kita mempunyai data harga dan data modal pada sebuah tabel, Lalu kita cek apakah harga melebihi dari modal yang diberikan kalau iya maka datanya akan muncul tetapi jika tidak maka dia tidak akan muncul. Rumusnya adalah seperti ini :

```
db.tabel.find( {$expr : { $gt : ["$harga","$modal"] } } ).pretty()
```

Keterangan :

- \$gt adalah operator disini kita dapat menggunakan operator lain sesuai kebutuhan.
- \$harga dan \$modal adalah sebuah nama di kolom tabel dan disini dapat kita panggil dengan menggunakan dolar (\$)

F. \$Next

Operator next adalah sebuah operator yang digunakan untuk menampilkan data di mulai data awal terlebih dahulu sampai data terakhir. Rumus dari next adalah seperti ini :

```
db.tabel.find().next()
```

Kalau dijalankan rumus di atas pada command prompt keluar data yang pertama tetapi jika kita jalankan lagi maka datanya tersebut akan memunculkan data pertama lagi. Untuk memunculkan data berikutnya kita memerlukan satu fungsi lagi yaitu menggunakan “const”, Const adalah sebuah penampung data yang apabila datanya sudah diisi maka tidak bisa dirubah kembali rumus pembuatan const adalah :

```
const nama-variabel = isi
```

Keterangan :

- Untuk nama-variabel bebas kita mau memberikan nama apa asal kita ingat nanti saat pemanggilannya
- Isi ini bisa kita isikan apa saja bebas asal sesuai dengan kebutuhan contoh kalau pada kasus next kita bisa mengisi db.tabel.find() . tabel diisi dengan tabel masing-masing ya

Kalau sudah dibuat sebuah variabelnya sekarang coba ketika “nama-variabel.next()” pada command prompt maka akan keluar data yang pertama, jika kita menuliskannya lagi maka akan muncul data seterusnya sampai data terakhir dan apabila sudah sampai data terakhir ingin menuliskannya lagi maka data tersebut tidak tampil.

G. Sort

Operator sort adalah sebuah operator yang digunakan untuk memilah data contoh kita data mengurutkan data dari A sampai Z atau sebaliknya, bisa juga untuk mengurutkan angka dari yang terkecil terlebih dahulu atau yang terbesar terlebih dahulu. Rumus dari sort adalah seperti ini :

```
db.tabel.find().sort({data:-1}).pretty()
```

Keterangan :

- Data adalah sebuah data yang akan di sortir
- -1 adalah pengurutan data berdasarkan descending atau menurun (selain -1 terdapat juga 1 yaitu ascending atau menaik)

H. Limit

Operator limit adalah sebuah operator yang berfungsi untuk memfilter data yang kita ingin tampilkan dengan jumlah yang telah di tentukan sebelumnya. Rumus dari sort adalah seperti ini :

```
db.tabel.find().limit(5).pretty()
```

Keterangan :

- Jumlah data yang ingin ditampilkan diisi di dalam kurung pada limit misalkan contoh di atas adalah 5.

I. \$Unset

Operator unset adalah sebuah operator yang berfungsi untuk menghapus data pada tabel. Misalkan kita salah memasukan sebuah data contoh pada suatu tabel barang data pertama berisi nama-barang lalu di data yang kedua berisi nama-barang, tipe. Lalu kita akan menghapus tipe di data yang ke dua agar data kita bisa sama dengan data yang pertama. Disini kita bisa menggunakan unset, untuk unset rumusnya adalah seperti ini :

```
db.tabel.updateMany({kriteria}, {$unset : {kriteria2} } )
```

Keterangan :

- Kriteria pertama adalah kriteria yang dipakai untuk update data.
- Kriteria2 adalah kriteria yang dipakai untuk menghapus data tersebut.

Contohnya

```
“db.barang.updateMany({}, { $unset : { tipe:tipe-barang } } )”
```

J. \$Rename

Operator rename adalah sebuah operator yang berfungsi untuk mengubah nama data pada tabel. Misalkan kita salah memasukan sebuah data contoh pada suatu tabel barang data pertama berisi nama-barang lalu di data yang kedua berisi nma-barang (salah pengetikan) lalu kita bisa merubah nma-barang yang salah pengetika menjadi nama-barang menggunakan operator rename. Untuk rumusnya adalah seperti ini:

```
db.tabel.updateMany({kriteria}, {$rename : {data: “data-baru”} } )
```

Keterangan :

- Data adalah sebuah nama data yang ada pada suatu tabel
- Data-baru adalah sebuah nama baru untuk merubah nama data awal

Contohnya :

```
“db.barang.updateMany({}, { $rename : { nma-barang:”nama-barang”} } } )”
```

K. \$Upsert

Operator upsert adalah sebuah operator yang digunakan untuk menambahkan suatu data jika data yang terupdate tidak di temukan, misalkan pada update disitu ada sebuah kriteria jika kriterianya itu tidak sesuai maka data tersebut akan disimpan secara otomatis. Untuk rumus upsert adalah seperti ini :

```
db.tabel.updateMany({kriteria}, { $set : { data : “isi” } }, { $upsert : true } )
```

L. \$ElemMacth

Operator elemMacth adalah sebuah operator yang digunakan untuk mencari data pada suatu data array. Misalkan kita mempunyai data array 80,90,100 lalu kita ingin mencari data yang lebih dari 80 dan kurang dari 100 kita disini bisa menggunakan operator elemMacth. Untuk rumus elemMacth seperti ini:

```
db.tabel.find( { data : { $elemMacth : { $gt : 80, $lt : 100 } } }).pretty()
```

Keterangan :

Diatas adalah rumus untuk mencari data array yang datanya lebih dari 80 dan kurang dari 100 pada data sebelumnya kan 80,90,100 untuk 80 dan 100 tidak masuk di kriteria tetapi 90 masuk kedalam kriteria sehinggal data array ini bisa dapat di tampilkan.

M. \$Push

Operator push adalah sebuah operator yang digunakan untuk menambahkan data pada sebuah data array. Misalkan kita mempunyai data array 70,80,90 lalu kita ingin menambahkannya menjadi 70,80,90,100 maka kita bisa menggunakan operator push. Untuk rumus push seperti ini :

```
db.tabel.updateOne({kriteria}, { $push : { data : isi } } )
```

Untuk menambahkan 1 data array

```
db.tabel.updateOne({kriteria}, {$push : {data : $each : [isi,isi,isi]} } )
```

Untuk menambahkan banyak data array

N. \$Pull

Operator pull adalah sebuah operator yang digunakan untuk menghapus data pada sebuah data array. Misalkan kita mempunyai data array 70,80,90 lalu kita ingin menghapus salah satunya menjadi 70,80 maka kita bisa menggunakan operator pull. Untuk rumus pull seperti ini :

```
db.tabel.updateOne({kriteria}, {$pull : {data : isi-yang-dihapus} } )
```

O. \$AddToSet

Operator addToSet adalah sebuah operator yang digunakan untuk menambahkan data pada sebuah data array tetapi jika datanya sudah ada pada array tersebut maka datanya tidak akan ditambahkan. Untuk rumus addToSet seperti ini :

```
db.tabel.updateOne({kriteria}, {$addToSet : {data : isi} } )
```

BAB V

A. Index

Index adalah suatu fungsi pada database yang berfungsi untuk mempercepat pengeksekusian sebuah data. Apabila kita menggunakan index pada suatu data di tabel kita, ketika menampilkan suatu data dengan berdasarkan index maka datanya tersebut akan menjadi lebih cepat munculnya dibandingkan dengan menggunakan kriteria yang bukan merupakan sebuah index. Untuk mengetahui apa saja index yang terdapat pada tabel kita disini kita misal menggunakan `getIndexes()` contoh :

```
“db.tabel. getIndexes()”
```

Cara untuk membuat sebuah indexnya adalah seperti ini :

```
“db.tabel.createIndex({data:1})”
```

Keterangan :

- Data adalah sebuah data pada tabel yang akan dijadikan index.
- 1 atau menaik dan -1 atau menurun adalah pengurutan datanya apabila nantinya data itu akan di tampilkan

Apabila kita membuat sebuah index banyak di sebuah data pada tabel maka datanya itu cepat dalam menampilkan sebuah data tetapi pada saat menghapus dan mengubah datanya maka akan membutuhkan waktu yang lama.

Untuk menambahkan index disini kita juga bisa memakai fungsi ini agar pembuatan index kita tidak satu – satu atau bisa langsung sekali banyak. Rumusnya adalah seperti ini :

```
“db.tabel.createIndex({data:1,data2:-1})”
```

Selanjutnya bagaimana caranya menghapus dari sebuah index. Rumusnya adalah seperti ini :

```
“db.tabel.dropIndex({data:1})”
```

Index mempunyai beberapa tipe salah satunya adalah unique. Unique adalah sebuah properties pada index mongodb yang bersifat tidak boleh sama atau hanya ada 1 pada tabel tersebut. Rumusnya adalah seperti ini :

```
“db.tabel.createIndex({data:1},{unique : true})”
```

Untuk membuat sebuah index tetapi indexnya itu hanya berlaku pada suatu data tertentu saja kita dapat menggunakan fungsi `partialFilterExpression`. Rumusnya adalah seperti ini :

```
“db.tabel.createIndex({data:1},{ partialFilterExpression : {kriteria} })”
```

Jika kita ingin membuat index tetapi membuat indexnya itu hanya bisa bertahan dalam kurun waktu tertentu saja kita bisa menggunakan `expireAfterSeconds`. Rumusnya adalah seperti ini :

```
“db.tabel.createIndex({data:1},{ expireAfterSeconds : 10 })”
```

Keterangan :

10 disini adalah waktu hidup dari index tersebut dalam satuan detik jadi jika ditulis 10 maka dalam kurun waktu 10 detik index tersebut akan hilang.

B. Geospatial

Geospatial adalah sebuah data tentang lokasi geografis dari suatu wilayah. Jika kita ingin mencari suatu wilayah tertentu pada suatu program kita dapat menggunakan fungsi ini. Untuk menjalankan sebuah geospatial kita disini kita harus menggunakan Geo Json, Geo Json adalah sebuah fungsi yang digunakan untuk menyimpan suatu titik wilayah tertentu. Tipe – tipe geo json ada banyak tetapi disini kita akan membahas 2 tipe json yaitu :

1. Point
Pencarian data dengan menggunakan titik koordinat suatu wilayah tertentu.
2. Polygon
Pencarian data dengan menggunakan beberapa titik koornidat suatu wilayah tertentu.

Sebelum masuk ke dalam kedua tipe ini coba kita buat sebuah database baru yang bernama “latihan6_nama-masing-masing” lalu buat sebuah collection “wilayah” pada mongodb compass. Tambahkan data seperti ini :

```
_id : ObjectId("5c10f80b291270089caf10e1")
nama : "Kampus Babastudio Permata Hijau"
lokasi : Object
  type : "Point"
  coordinates : Array
    0 : "106.7816441"
    1 : "-6.2205147"
```

Untuk yang pertama kita akan mencari data dengan menggunakan geo json tipe point yang rumusnya adalah seperti ini :

```
“db.tabel.find({lokasi : {$near { $geometry : { type : “Point”, coordinates : [106.7816441,-6.2205147] } }}) pretty()”
```

Keterangan :

- Lokasi adalah nama data pada sebuah tabel yang menyimpan type dan titik coordinates.
- \$near adalah sebuah query dari geospatial yang berfungsi untuk menentukan titik kordinat dari suatu objek tertentu.
- \$geometry adalah rumus dari geospatial yang harus di pakai kalau untuk mau menampilkan sebuah objek berdasarkan titiknya
- Type : “Point” adalah geo json tipe point
- Coordinates adalah nama data yang menyimpan titik coordinates

Selanjutnya untuk menjalankan kode diatas pada data yang menyimpan kordinasi harus mempunyai suatu index dengan properties “2dsphere”.untuk pembuatannya rumusnya seperti ini :

```
db.tabel.createIndex({"lokasi" : "2dsphere"})
```

Untuk geo json tipe polygon kita harus membuat sebuah titik wilayah terlebih dahulu coba kita buka google maps.



Klik menu

-  Notifikasi
-  Bagi lokasi
-  Tempat Anda
-  Kontribusi Anda
-  Linimasa Anda
-  Bagikan atau sematkan peta
-  Cetak

Klik tempat anda

BERLABEL

DISIMPAN

DIKUNJUNGI

PETA



Peta tanpa judul

13 Desember 2018



Peta tanpa judul

13 Desember 2018



Peta tanpa judul

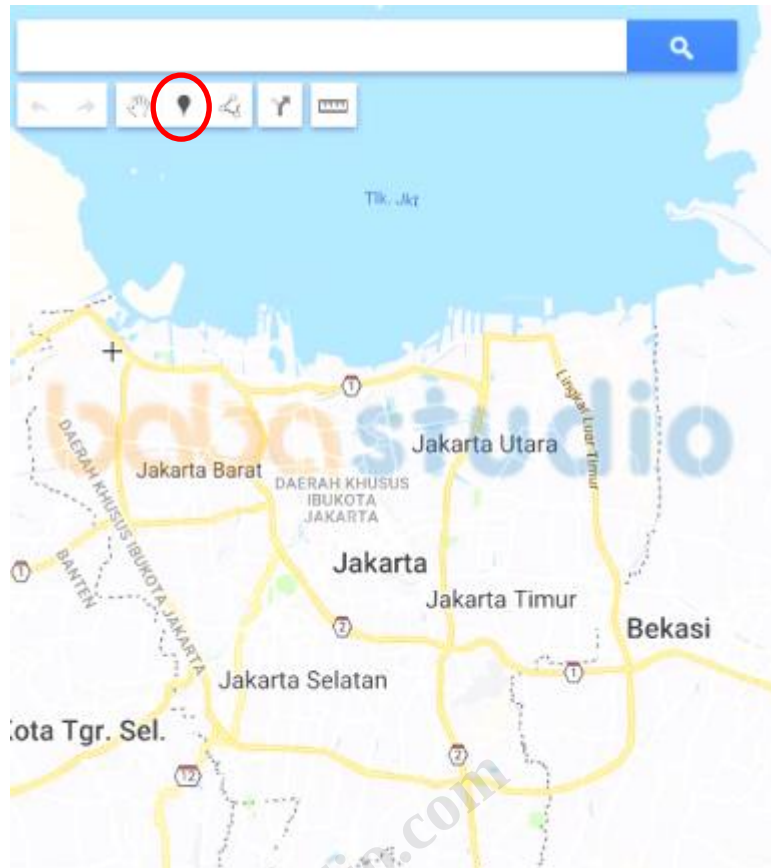
12 Desember 2018

LIHAT SEMUA PETA ANDA

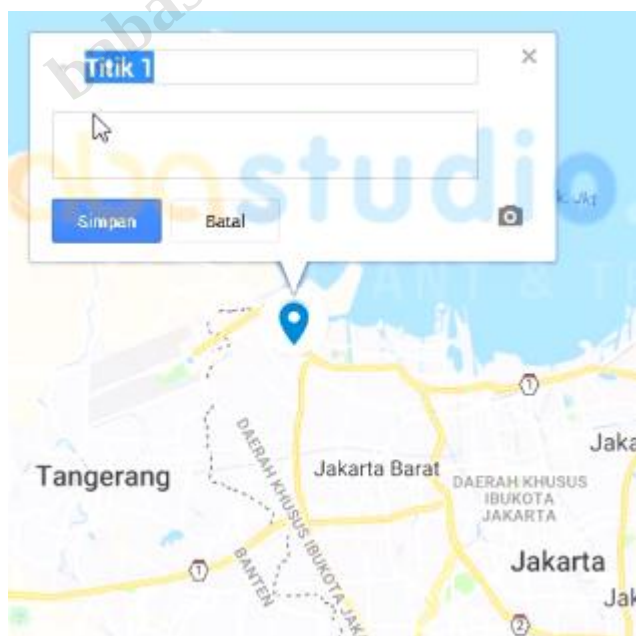
BUAT PETA

Klik peta lalu buat peta

Selanjutnya coba cari daerah ibu kota Jakarta pada maps



Kita buat titik pada suatu wilayah klik pada bulatan yang ada di foto.



Pilih titiknya disini beri nama titik 1



Buat 4 titik seperti ini



Lalu kita berikan sebuah garis, kita pilih titik pertama dan arahkan ke titik yang kedua seperti ini.



Lalu buat garisnya sampai berbentuk seperti ini



Sehingga di menjadi sebuah polygon, sekarang kita membuat sebuah variabel sesuai dengan titik – titik yang telah kita tentukan diatas pada command prompt :

```
> const p1 = [106.72618,-6.11225]
> const p2 = [106.75983,-6.28973]
> const p3 = [106.95895,-6.31089]
2018-12-13T13:53:51.796+0700 E QU
@ (shell) :1:7
> const p4 = [106.95895,-6.31089]
> const p5 = [106.92187,-6.10201]
```

Untuk rumus geo json bertipe polygon rumusnya seperti ini:

```
"db.tabel.find({lokasi : {$geoWithin : { $geometry : { type : "Polygon",
coordinates : [p1,p2,p3,p4,p1] } } }).pretty()"
```

Keterangan :

- \$geoWithin adalah memilih suatu data berdasarkan beberapa titik kordinat tertentu
- [p1,p2,p3,p4,p1] adalah titik – titik kordinates yang sudah di tentukan sebelumnya. P1 di belakang adalah penutup jadi dari p1 kita Tarik garis ke p2,dari p2 ke p3, dari p3 ke p4, dari p4 ke p1 lagi untuk menutup garisnya supaya menjadi sebuah polygon.

C. Capped Collection

Capped Collection adalah sebuah fungsi yang digunakan untuk membuat batasan yang ada pada saat membuat tabel. Misalkan ada suatu data yang telah di berikan batas oleh capped collection yaitu data A, data B, data C. Apabila kita menambahkan sebuah data baru maka data yang terlama akan terganti dengan data yang baru. Pada fungsi ini juga bisa

membatasi ukuran memori dari suatu data. Untuk penulisan rumusnya seperti ini :

```
“db.createCollection(“contoh”, {capped:true , size : 10000, max :3 })”
```

Keterangan :

- Contoh adalah nama tabel yang akan dibuat
- Capped : true untuk mengaktifkan fungsi capped collection
- Size adalah ukuran memori dari sebuah tabelnya (10000 adalah ukuran dalam bentuk byte)
- Max adalah maksimal data yang bisa tersimpan pada tabel kita

D. Mongodb Atlas

Mongodb atlas adalah sebuah GUI atau bentuk tampilan yang disediakan oleh mongodb. Kalau sebelumnya kita sudah mengenal mongo compass yang berbentuk software desktop sekarang kita akan belajar tentang mongodb atlas yaitu sebuah server penyimpanan data secara online oleh mongodb. Coba kita masuk ke dalam browser kita masing-masing :



Ketikan mongo db atlas

MongoDB Atlas

<https://www.mongodb.com/cloud/atlas>

Dec 12, 2018 - MongoDB Atlas delivers the world's leading database for modern applications as a fully automated cloud service engineered and run by the ...

MongoDB Atlas Pricing

With MongoDB Atlas, straightforward hourly pricing ...

MongoDB Atlas FAQ

FAQ Table of Contents. What is MongoDB Atlas? Where is ...

More results from mongodb.com »

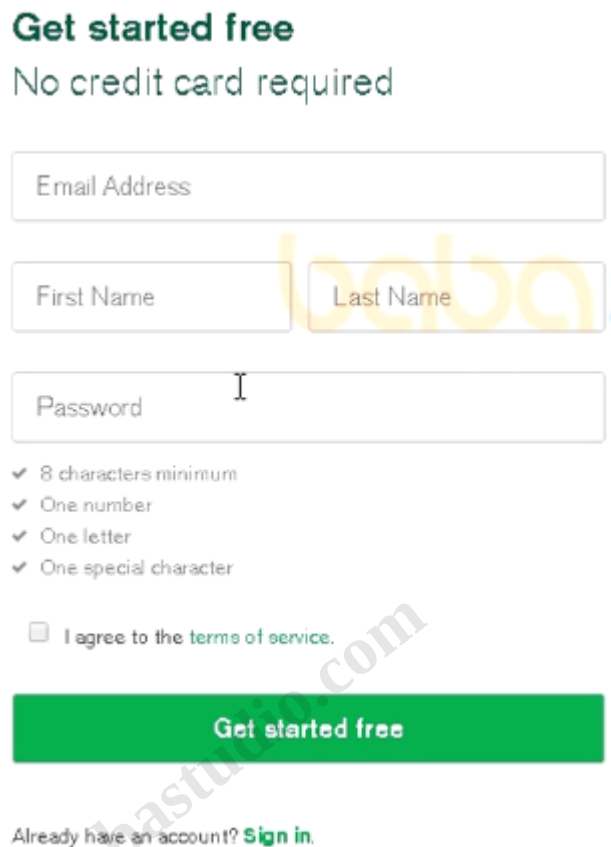
Migrate to MongoDB Atlas

MongoDB Atlas delivers the flexibility, scalability, and ...

Cloud Provider Snapshots

Atlas Cloud Provider Snapshots provide localized backup ...

Lalu pilih yang ini, maka kita akan masuk ke dalam website mongoddb yang menyediakan server mongoddb secara online atau mongoddb atlas. Coba kita scroll ke bawah maka akan tampil seperti ini :



The image shows a registration form for MongoDB Atlas. At the top, it says "Get started free" in green, followed by "No credit card required" in a smaller green font. Below this are four input fields: "Email Address", "First Name", "Last Name", and "Password". The "Password" field has a cursor and is followed by four requirements: "8 characters minimum", "One number", "One letter", and "One special character", each preceded by a green checkmark. Below the password field is a checkbox labeled "I agree to the terms of service." with a link to "terms of service". At the bottom of the form is a large green button labeled "Get started free". Below the button, it says "Already have an account? Sign in." with "Sign in" as a link. A large, semi-transparent watermark "babastudio.com" is visible across the center of the form.

Get started free
No credit card required

Email Address

First Name Last Name

Password

- ✓ 8 characters minimum
- ✓ One number
- ✓ One letter
- ✓ One special character

☐ I agree to the [terms of service](#).

Get started free

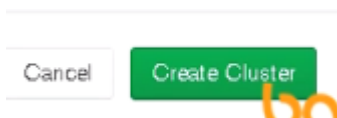
Already have an account? [Sign in](#).

Lalu coba kita isi form ini. Untuk password minimal 8 karakter, 1 angka, 1 huruf besar , dan 1 special karakter atau symbol. Kalau sudah coba klik tombol get started free, maka kita akan langsung di arahkan ke cloud mongoddb.

★ recommended region ⓘ

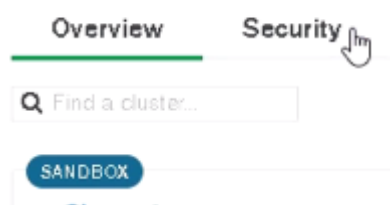
NORTH AMERICA	EUROPE	AUSTRALIA
N. Virginia (us-east-1) ★ FREE TIER AVAILABLE	Ireland (eu-west-1) ★	Sydney (ap-southeast-2) ★
Ohio (us-east-2) ★	London (eu-west-2) ★	ASIA
N. California (us-west-1)	Paris (eu-west-3) ★	Tokyo (ap-northeast-1) ★
Oregon (us-west-2) ★	Frankfurt (eu-central-1) ★ FREE TIER AVAILABLE	Seoul (ap-northeast-2)
Montreal (ca-central-1)	SOUTH AMERICA	Singapore (ap-southeast-1) ★ FREE TIER AVAILABLE
	Sao Paulo (sa-east-1)	Mumbai (ap-south-1) FREE TIER AVAILABLE

Pada recommended region kita pilih yang server Singapore, kita pilih Singapore karna posisinya lebih dekat dari Indonesia (dengan semakin dekatnya server pada saat kita mau mengakses suatu website maka loadingnya juga akan cepat) dan dia bisa gratis. Kalau sudah di pilih regionnya sekarang coba scroll ke bawah lalu klik tombol Create Cluster.



Setelah kita create cluster maka kita akan dibawa ke tempat cluster itu tersimpan. sekarang kita akan membuat sebuah user yang berfungsi untuk pengamanan database kita agar tidak sembarang orang yang masuk ke dalam mongodb. Caranya :

Clusters



Add New User

SCRAM Authentication
SCRAM is MongoDB's default authentication method.

Enter username
e.g. new-user_31

Enter password SHOW
[Autogenerate Secure Password](#)

User Privileges

Atlas admin **Read and write to any database** Only read any database Select Custom Role

[Add Default Privileges](#)

☐ Save as temporary user Cancel Add User

Klik security. Lalu akan muncul popup add new user, coba diisi untuk username, password, dan user privileges. Pada user privileges terdapat 4 macam yaitu Admin, Read and write to any database, only read any database dan select custom role (pilih sesuai kebutuhan kalau disini babastudio merecommmed user admin).

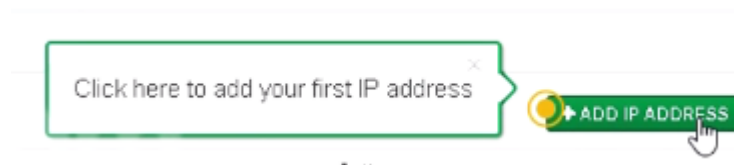
1. Admin
User privileges admin adalah sebuah user yang dapat memanipulasi sebuah data
2. Read and write to any database
User privileges Read and write to any database adalah sebuah user yang dapat membaca dan menulis saja pada mongodb
3. Only read any database
User privileges Only read any database adalah sebuah user yang hanya dapat membaca saja pada mongodb
4. Custom role
User custom role ini adalah sebuah perpaduan antara ketiga user.

Kemudian kita klik add user .

Clusters



Kemudian kita klik IP Whitelist



Lalu kita add ip address

A screenshot of the 'Add Whitelist Entry' dialog box. The dialog has a title bar with a close button (X). The main content area has the title 'Add Whitelist Entry' and a subtitle 'Add a whitelist entry using either CIDR notation or a single IP address. Learn more.' Below the subtitle are two buttons: 'ADD CURRENT IP ADDRESS' and 'ALLOW ACCESS FROM ANYWHERE'. There are two input fields: 'Whitelist Entry:' with the placeholder 'Enter IP Address or CIDR Notation' and 'Comment:' with the placeholder 'Optional comment describing this entry'. At the bottom left is a checkbox labeled 'Save as temporary whitelist'. At the bottom right are two buttons: 'Cancel' and 'Confirm'.

Maka akan muncul pop up seperti ini kita isi whitelist & commentnya, untuk whitelist diisi dengan ip pengguna database. Kalau sudah diisi kita tekan tombol confirm.

Connect to Atlas

Follow this checklist to get started.

75% 

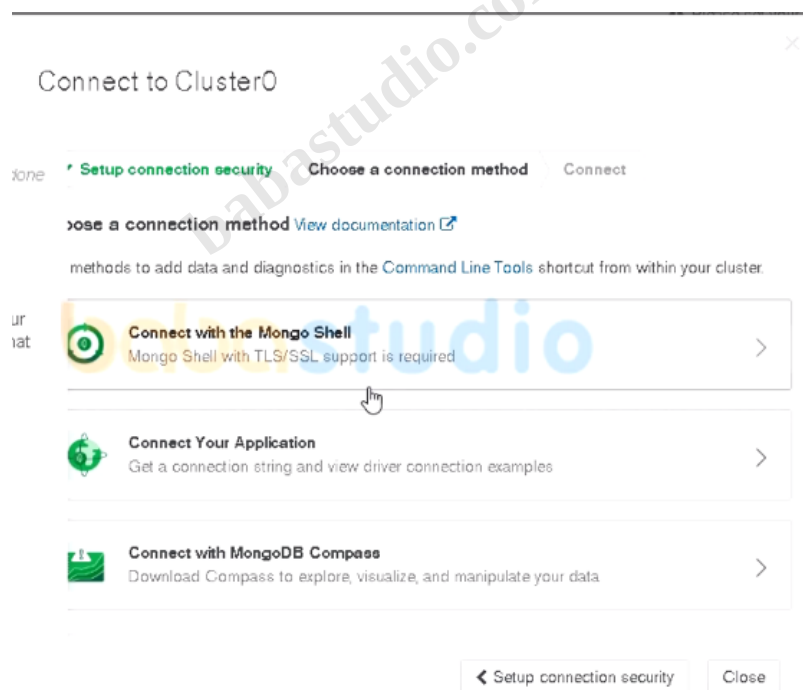
- ☒ Build your first cluster
- ☒ Create your first database user
- ☒ Whitelist your IP address
- ☐ Connect to your cluster

No Thanks

 Get Started ¹

System Status: All Green

Coba kita klik get started kembali maka disini telah complete 75% lalu kita klik yang connect to your cluster.



Disini coba kita pilih yang connect with the mongo shell

Setup connection security Choose a connection method Connect

1 If you are not running MongoDB 4.0.4+ with TLS/SSL, click below to download the latest Mongo Shell

Windows Mac OS X Other Operating Systems

Download

Or fetch with this URL:

https://downloads.mongodb.org/win32/mongodb-shell-win32-x86_64-2008f COPY

2 Connect via the Mongo Shell

View detailed instructions

Short SRV connection string (shell 3.6+) Standard connection string (shell 3.4+)

```
mongo "mongodb+srv://cluster0-9wm0a.mongodb.net/test" --username student
```

COPY

You will be prompted for the password for the *student* user. Please ensure you are using a MongoDB 3.6.1+ shell with TLS/SSL.

Selanjutnya coba kita copykan script yang ada di langkah yang ke 2. Lalu coba kita jalankan scriptnya ke commad prompt.

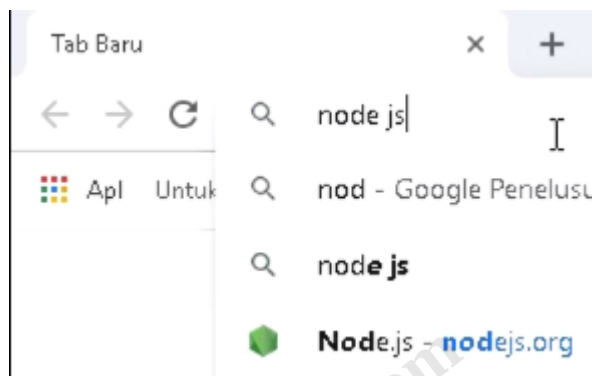
```
C:\Program Files\MongoDB\Server\4.0\bin>mongo "mongodb+srv://cluster0-9wm0a.mongodb.net/test" --username student
MongoDB shell version v4.0.4
Enter password: █
```

Langkah diatas apabila kita copykan data tersebut ke dalam mongodb maka akan dimintakan sebuah password yang telah di buat sebelumnya. Jika sudah terkoneksi kita dapat menggunakannya seperti bisa kita menggunakan command prompt mongodb.

BAB VI

A. Node js

Nodejs adalah perangkat lunak yang didesain untuk mengembangkan aplikasi berbasis web dan ditulis dalam sintaks bahasa pemrograman JavaScript. Pada bab kali ini kita akan mengimplementasikan nodejs pada mongodb yang telah kita pelajari. Sebelum kita mengimplementasikannya coba kita install terlebih dahulu node jsnya, coba kita buka browser lalu coba ketikan nodejs seperti ini :



Lalu pilih yang ini



Maka akan muncul seperti ini



Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Download for Windows (x64)

10.14.2 LTS

Recommended For Most Users

11.5.0 Current

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#) [Other Downloads](#) | [Changelog](#) | [API Docs](#)

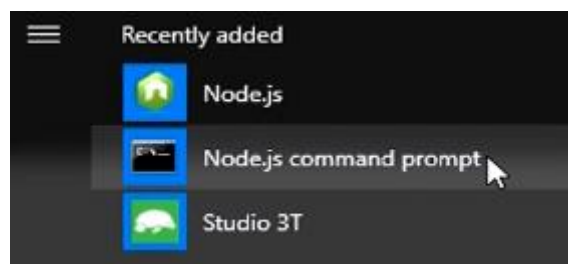
Or have a look at the [Long Term Support \(LTS\) schedule](#).

Sign up for [Node.js Everywhere](#), the official Node.js Monthly Newsletter.

Pilih yang recommended for most user maka akan mendownloadnya secara otomatis. Kalau sudah terdownload coba kita buka.



Untuk penginstalannya cukup tekan next > change > next > change > finish. Kalau sudah terinstall sekarang coba klik start tuliskan Node.js command prompt.



Selanjutnya coba download file yang ada di referensi yang bernama latihan node lalu taruh di dalam folder masing – masing. Kalau sudah coba kita buka Node.js command prompt ketika direktori dimana kita menaruh file latihan nodejs nya (kalau misalkan beda drive direktori kita dapat menggunakan “cd /” enter lalu ketikan drive direktori misalkan untuk data D kita dapat menuliskan “d:” enter lalu tulislah direktorinya. Lalu jalankan script seperti ini

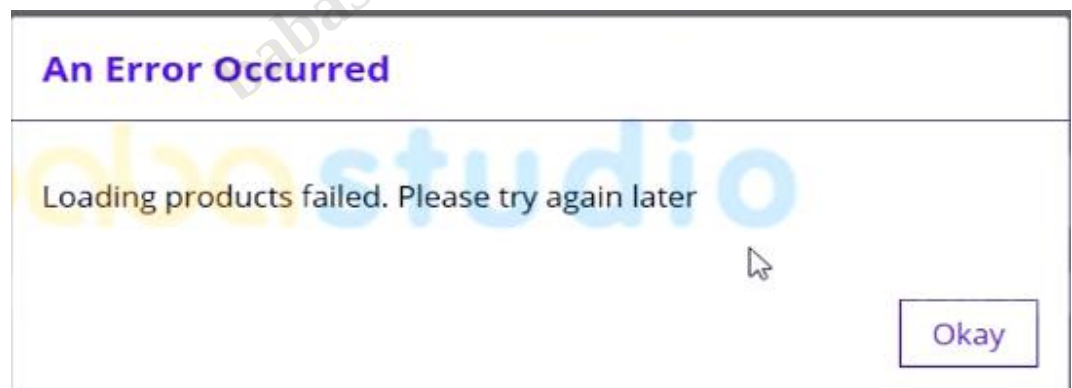
```
F:\Murid\baba_mongodb\latihan_nodejs>npm install
> uglifyjs-webpack-plugin@0.4.6 postinstall F:\Murid\baba_mongodb\latihan_nodejs\node_modules\uglifyjs-webpack-plugin
> node lib/post_install.js
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
added 1287 packages from 745 contributors and audited 14755 packages in 104.15s
found 3 vulnerabilities (2 low, 1 high)
  run `npm audit fix` to fix them, or `npm audit` for details
```

Keterangan :

- Npm install adalah sebuah paket dari javascript yang nanti apabila di install paket data – data yang sebelumnya tersembunyi di akan tampil.

```
F:\Murid\baba_mongodb\latihan_nodejs>npm start
> mongodb-demo@0.1.0 start F:\Murid\baba_mongodb\latihan_nodejs
> react-scripts start
```

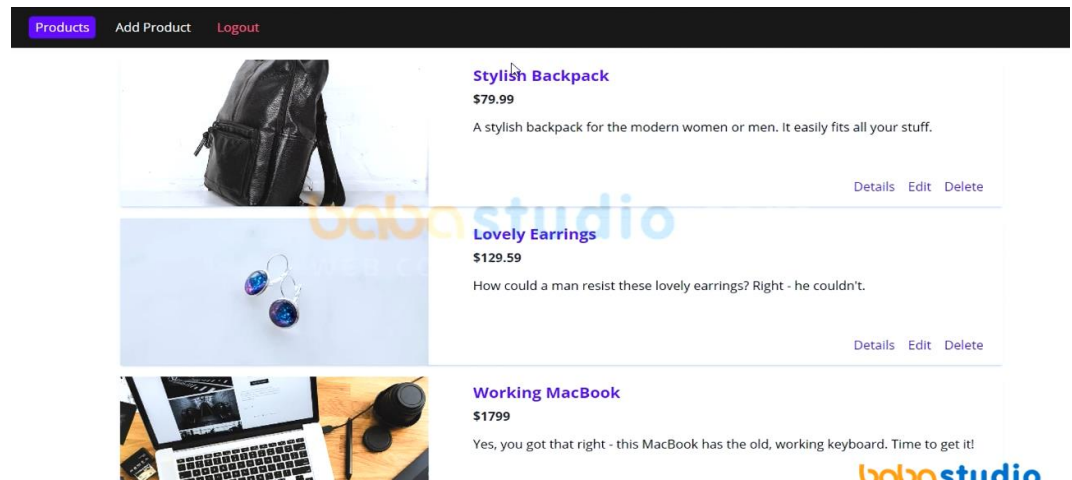
Lalu ketikan npm start untuk menjalankan websitenya.



maka websitenya akan error ya karena dia tidak bisa terbaca oleh server. Untuk mengatasi masalah seperti ini coba kita buka node.js command prompt yang kedua lalu kita masuk lagi di command prompt ke folder tempat dimana latihan_nodejs berada. Sekarang coba ketikan seperti ini :

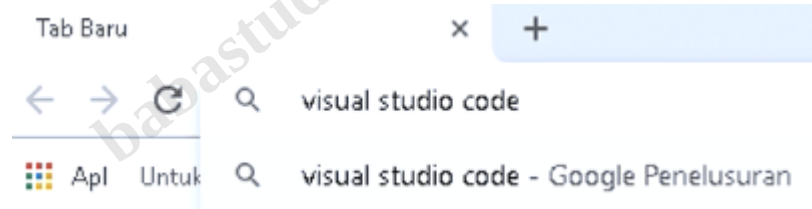
```
F:\Murid\baba_mongodb\latihan_nodejs>npm run start:server
> mongodb-demo@0.1.0 start:server F:\Murid\baba_mongodb\latihan_nodejs
> node ./backend/app.js
```

Setelah menjalankan sebuah servernya coba kita kembali ke browser kita lalu kita refresh halaman website latihan_nodejs nya. Maka tampilannya akan seperti ini :



B. Virtual Studio Code

Visual studio code adalah editor webstie yang dapat mengubah / membuat beberapa data web seperti js, php, html , dll. Coba kita buka browser lagi untuk mendownload Visual studio code :



Lalu kita pilih yang ini

[Visual Studio Code - Code Editing. Redefined](https://code.visualstudio.com/)

<https://code.visualstudio.com/>

Visual Studio Code is a code editor redefined and optimized for building and debugging modern web and cloud applications. Visual Studio Code is free and ...

Download Visual Studio Code

Download Visual Studio Code to experience a redefined code ...

Documentation for Visual ...

Intro Videos - Basic Editing - Node.js - Tutorial - IntelliSense - C++

Updates

There are a number of significant updates in this version that we ...

Insiders

Download Visual Studio Code Insiders. Get the latest release ...

Visual Studio Code FAQ

Why VS Code - License - Portable Mode - ...

Setting up Visual Studio Code

Get Visual Studio Code up and running.

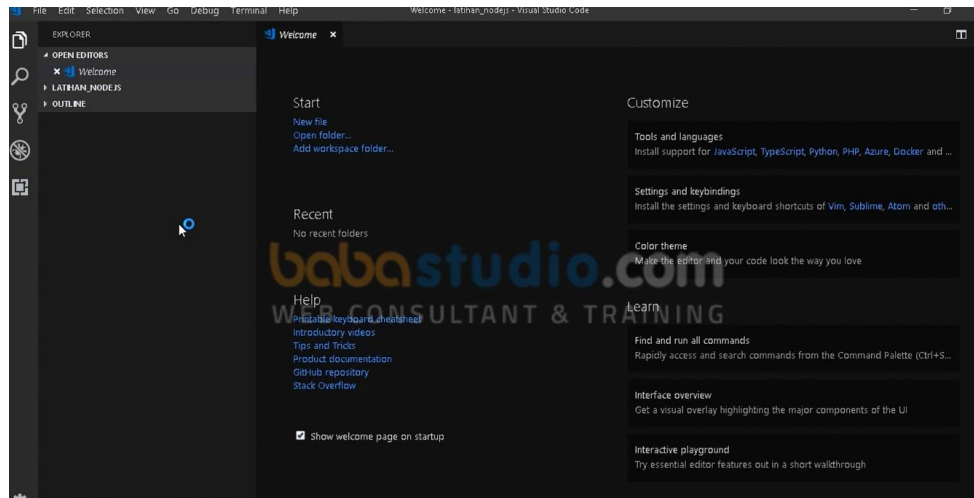
Setelah di buka maka tampilannya akan seperti ini :



Kita pilih yang download for windows. Kalau sudah selesai di download coba kita install aplikasinya



Untuk penginstallnya tinggal kita klik next next saja ya. Kalau sudah terinstall maka akan tampak seperti ini :



C. Npm mongodb

Untuk mengimplementasikan node js ke mongodb disini kita harus menginstall npm mongodb dahulu untuk penginstallannya untuk rumusnya seperti ini.

```
“npm install mongodb --save”
```

D. Membuat koneksi nodejs ke mongodb

Untuk membuat koneksi ke mongodb kita harus membuat sebuah user baru di mongodb atlas sebagai Read and Write any database cara pembuatannya sama dengan cara membuat user student(admin) untuk usernya kita beri nama “mongodb”. Buka aplikasi visual studio code lalu kita open file yang bernama app.js yang berada pada folder backend :

```

1  const path = require('path');
2
3  const express = require('express');
4  const bodyParser = require('body-parser');
5
6  const productRoutes = require('./routes/products');
7  const authRoutes = require('./routes/auth');
8
9  const app = express();
10
11 app.use(bodyParser.json());
12 app.use('/images', express.static(path.join('backend/images')));
13
14 app.use((req, res, next) => {
15   // Set CORS headers so that the React SPA is able to communicate with this server
16   res.setHeader('Access-Control-Allow-Origin', '*');
17   res.setHeader(
18     'Access-Control-Allow-Methods',
19     'GET, POST, PUT, PATCH, DELETE, OPTIONS'
20   );
21   res.setHeader('Access-Control-Allow-Headers', 'Content-Type');
22   next();
23 });
24
25 app.use('/products', productRoutes);
26 app.use('/', authRoutes);
27
28 app.listen(3100);
29

```

```

1  const path = require('path');
2
3  const express = require('express');
4  const bodyParser = require('body-parser');
5  const mongodb = require('mongodb').MongoClient;

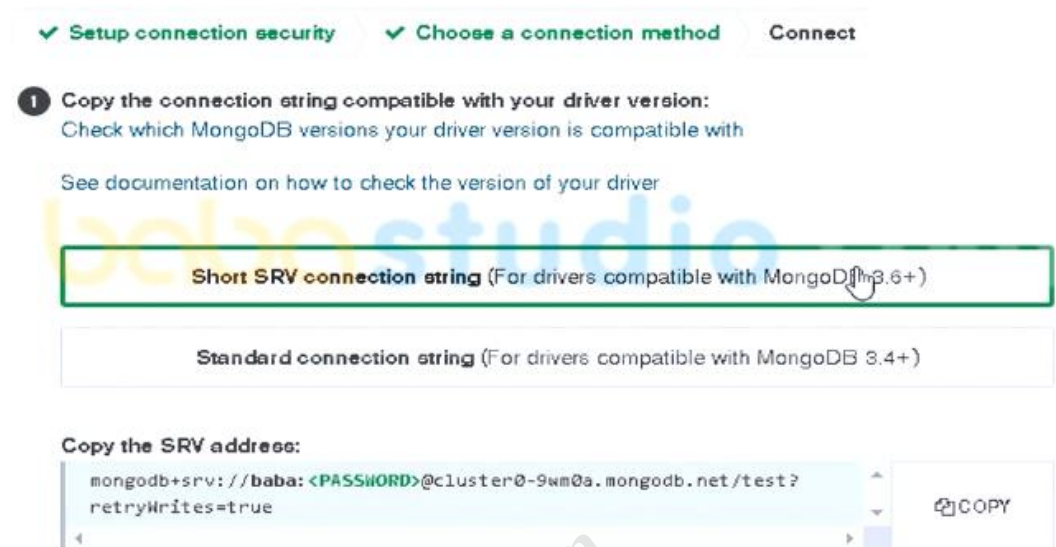
```

Pada baris ke 5 kita tambahkan seperti ini. Kita membuat variabel untuk memanggil mongodb dan kita gunakan untuk pengguna. Sekarang coba kita masuk kembali ke mongodb atlas.

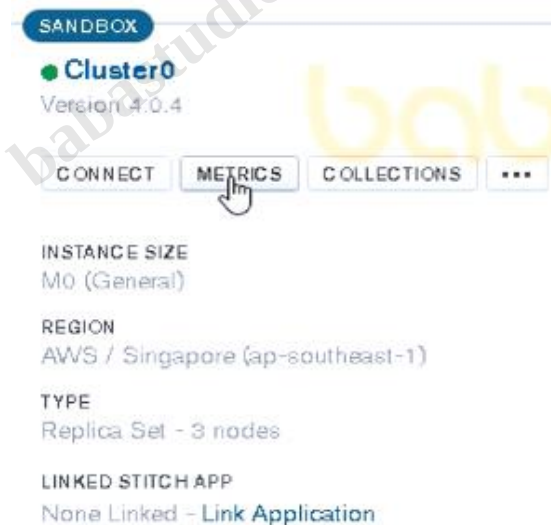
The screenshot shows the MongoDB Atlas interface. On the left, the 'Clusters' sidebar is open, showing 'Cluster0' (Version 4.0.4) with a 'CONNECT' button. The main area is titled 'Connect to Cluster0' and includes a 'Setup connection security' status. Below this, there are three connection methods: 'Connect with the Mongo Shell', 'Connect Your Application', and 'Connect with MongoDB Compass'. At the bottom right, there are buttons for '< Setup connection security' and 'Close'.

Klik connect maka akan tampil connect to cluster lalu disini kita pilihnya buka mongo shell lagi tetapi disini kita pilih yang connect your application.

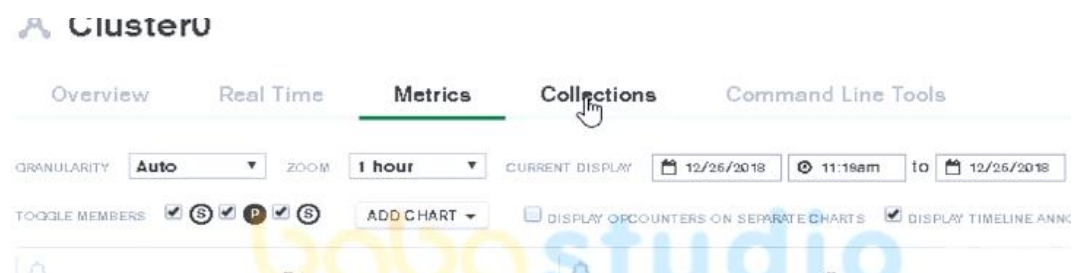
Connect to Cluster0



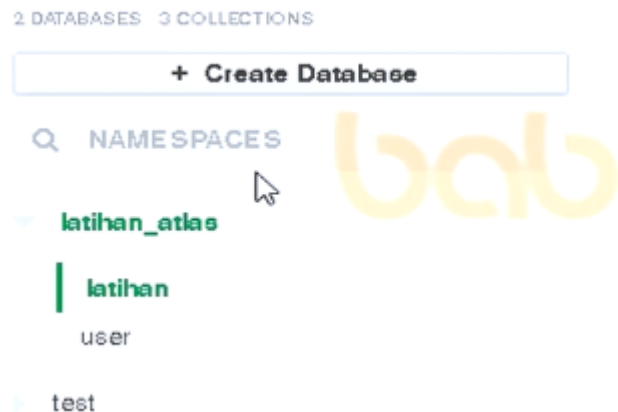
Jika muncul seperti ini kita klik yang short srv lalu kita copy the srv address. Selanjutnya kita buat database baru dengan cara klik metrics :



Lalu klik collection



Untuk membuat databasenya kita bisa klik create database.



Buat database dengan nama shop dan collections product lalu kita klik create. Lalu tambahkan script ini di bawah `app.use('/', authRouter)`.

```
26 app.use('/products', productRoutes);
27 app.use('/', authRoutes);
28
29 mongodb.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=true')
30 .then(client => {
31   console.log('Connected!');
32   client.close();
33 })
34 .catch(err => {
35   console.log(err);
36 });
```

Keterangan :

- Pada line ke 29 `srv://nama-user-yang sudah dibuat:password-usernya`
- Pada line ke 29 `shop` adalah nama databasenya
- Ini dari script di atas adalah kita membuat sebuah koneksi yang apabila kita berhasil mengkoneksikan database maka akan muncul pesan "Connected!" dan apabila gagal maka akan muncul pesan error.

```
F:\Murid\baba_mongodb\latihan_nodejs>npm run start:server
> mongodb-demo@0.1.0 start:server F:\Murid\baba_mongodb\latihan_nodejs
> node ./backend/app.js
(node:25588) DeprecationWarning: current URL string parser is deprecated
e the new parser, pass option { useNewUrlParser: true } to MongoClient.
Connected!
```

Maka apabila kita jalankan kembali servernya maka akan muncul pesan connected.

E. Menambahkan sebuah product

Untuk menambahkan sebuah product kita buka terlebih dahulu `product.js` yang berada di folder `routes` di `backend`. Kalau sudah di buka pada `app.js` variabel `mongodb` yang barusan kita buat di cut atau potong

kita taruh di dalam product.js dan kita pisahkan antara pemanggilan mongodb dengan clientnya makanya jadinya seperti ini :

```
1  const Router = require('express').Router;
2  const mongodb = require('mongodb');
3
4  const MongoClient = mongodb.MongoClient;
5
```

Lalu kita cut atau potong lagi di app.js baris ke 29 sampai 36, lalu di pastekan seperti gambar ini :

```
79  // Add new product
80  // Requires logged in user
81  router.post('', (req, res, next) => {
82    const newProduct = {
83      name: req.body.name,
84      description: req.body.description,
85      price: parseFloat(req.body.price), // store this as 128bit decimal in MongoDB
86      image: req.body.image
87    };
88    mongodb.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=true')
89      .then(client => {
90        console.log('Connected!');
91        client.close();
92      })
93      .catch(err => {
94        console.log(err);
95      });
96    res.status(201).json({ message: 'Product added', productId: 'DUMMY' });
97  });
```

Kita buat variabel lagi untuk mengkonversi kata menjadi sebuah angka decimal:

```
1  const Router = require('express').Router;
2  const mongodb = require('mongodb');
3
4  const MongoClient = mongodb.MongoClient;
5  const Decimal128 = mongodb.Decimal128;
```

Selanjutnya kita ubah baris 86 menjadi

```
80  // Add new product
81  // Requires logged in user
82  router.post('', (req, res, next) => {
83    const newProduct = {
84      name: req.body.name,
85      description: req.body.description,
86      price: Decimal128.fromString(req.body.price.toString()),
87      image: req.body.image
88    };
89  });
```

Karna sebelumnya price ini berupa kata maka kita konversi ke decimal. Lalu kita edit baris 88 sampai 97 dengan script seperti ini

```

89 MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites
90 .then(client => {
91   client
92     .db()
93     .collection('product')
94     .insertOne(newProduct)
95     .then(result =>{
96       console.log(result);
97       client.close();
98       res.status(201)
99       .json({ message : 'Product Added', productId: result.insertedId });
100     })
101     .catch(err => {
102       console.log(err);
103       client.close();
104       res.status(500)
105       .json({ message : 'An Error Occurred' });
106     });
107   })
108   .catch(err => {
109     console.log(err);
110   });
111 });

```

Lalu coba kita restart server npmnya dengan cara tekan “CTRL + C” pada nodejs command prompt lalu jalankan lagi servernya.

Title	Cangkir Cantik
Price	50000
Image URL	localhost:3000/backend/images/product-cangkir
Description	Cangkir cantik babastudio punya
Create Product	

Lalu kita tambahkan datanya, maka apabila kita lihat pada nodejs command prompt maka akan tampil seperti ini :

```
npm
  promoteValues: true,
  promoteBuffers: false },
  length: 245,
  requestId: 620582042,
  responseTo: 9,
  opCode: 1,
  fromCompressed: undefined,
  responseFlags: 8,
  cursorId: Long { _bsontype: 'Long', low: 0, high: 0 },
  startingFrom: 0,
  numberReturned: 1,
  documents: [ [Object] ],
  cursorNotFound: false,
  queryFailure: false,
  shardConfigStale: false,
  awaitCapable: true,
  promoteLongs: true,
  promoteValues: true,
  promoteBuffers: false,
  index: 220,
  hashedName: 'a2ebbfd1a590176b1e598cf59130383c72d2c0b7' },
  ops:
    [ { name: 'Cangkir Cantik',
      description: 'Cangkir cantik babastudio punya',
      price: [Decimal128],
      image: 'localhost:3000/backend/images/product-cangkir',
      id: 5c237db3cc82926f70f1758b } ],
  insertedCount: 1,
  insertedId: 5c237db3cc82926f70f1758b }
```

Kalau tampil seperti ini maka kita telah berhasil menambahkan datanya.

F. Menampilkan product

Untuk menampilkan product coba kita copykan terlebih dahulu data yang sebelumnya kita sudah ketikan:

```
89 MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=
90 .then(client => {
91   client
92   .db()
93   .collection('product')
94   .insertOne(newProduct)
95   .then(result =>{
96     console.log(result);
97     client.close();
98     res.status(201)
99     .json({ message : 'Product Added', productId: result.insertedId });
100   })
101   .catch(err => {
102     console.log(err);
103     client.close();
104     res.status(500)
105     .json({ message : 'An Error Occurred' });
106   });
107 })
108 .catch(err => {
109   console.log(err);
110 });
```

Kita taruh di dalam router.get('/') di baris ke 59, sebelum kita taruh di nonaktifkan atau kita komentar terlebih dahulu script yang ada di dalam router get('/') itu seperti ini :

```

// Get list of products products
router.get('/', (req, res, next) => {
  // Return a list of dummy products
  // Later, this data will be fetched from MongoDB
  // const queryPage = req.query.page;
  // const pageSize = 5;
  // let resultProducts = [...products];
  // if (queryPage) {
  //   resultProducts = products.slice(
  //     (queryPage - 1) * pageSize,
  //     queryPage * pageSize
  //   );
  // }
  // }
  res.json(resultProducts);
});

```

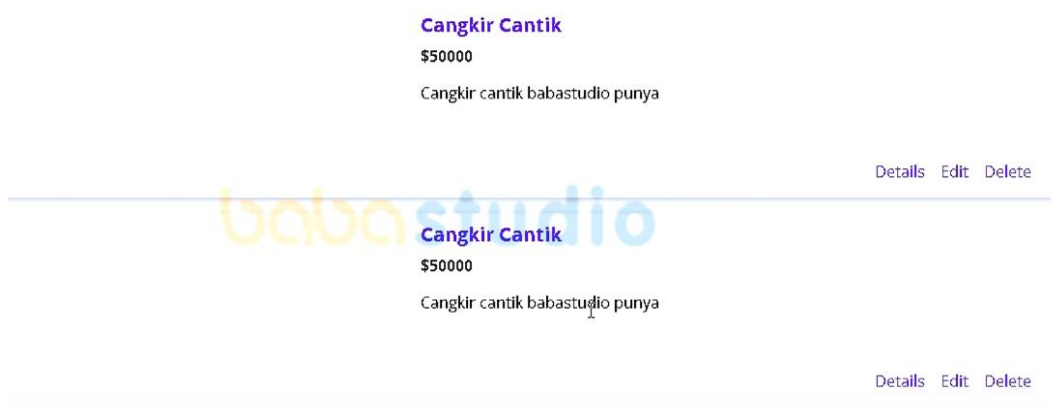
Kita paste dan kita edit sehingga nanti hasilnya seperti ini :

```

72 MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=
73 .then(client => {
74   const products = [];
75   client
76   .db()
77   .collection('product')
78   .find()
79   .forEach(Dataprodukt => {
80     // console.log(Dataprodukt);
81     // return Dataprodukt;
82     Dataprodukt.price = Dataprodukt.price.toString();
83     products.push(Dataprodukt);
84   })
85   .then(result => {
86     client.close();
87     res.status(200)
88     .json([
89       ]
90     )
91     .catch(err => {
92       console.log(err);
93       client.close();
94       res.status(500)
95       .json({ message : 'An Error Occurred' });
96     });
97   });
98   .catch(err => {
99     console.log(err);
100     res.status(500)
101     .json({ message : 'An Error Occurred' });
102   });
103   //res.json(resultProducts);
104 });

```

Kalau berhasil maka akan tampil seperti ini :



Jika gambarnya tidak muncul sama seperti yang di gambar coba kita buka mongoddb atlas kita edit data untuk data cangkir cantik ini menjadi seperti ini.



Kita buah data imagenya menjadi seperti itu. Lalu klik update maka datanya akan tampil seperti ini :



G. Menampilkan product detail

Untuk menampilkan product detail kita buat sebuah variabel baru terlebih dahulu untuk menampung data object id nya

```

3
4  const MongoClient = mongodb.MongoClient;
5  const Decimal128 = mongodb.Decimal128;
6
7  const ObjectId = mongodb.ObjectId;

```

Lalu kita ketikkan script ini untuk menampilkan product detailnya :

```
106 // Get single product
107 router.get('/:id', (req, res, next) => {
108   MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=
109   .then(client => {
110     client
111     .db()
112     .collection('product')
113     .findOne({ _id: new ObjectId(req.params.id) })
114     .then(Dataproduct =>{
115       Dataproduct.price = Dataproduct.price.toString();
116       res.status(200)
117       .json(Dataproduct);
118     })
119     .catch(err => {
120       console.log(err);
121       res.status(500)
122       .json({ message : 'An Error Occurred' });
123     });
124   })
125   .catch(err => {
126     res.status(500)
127     .json({ message : 'An Error Occurred' });
128   });
129 });
130
```

Kita restart kembali npm server lalu jika di jalankan maka hasilnya adalah

Cangkir Cantik

50000



Cangkir cantik babastudio punya

H. Mengubah product

Untuk mengubah product coba kita tuliskan script ini :

```
166 // Edit existing product
167 // Requires logged in user
168 router.patch('/:id', (req, res, next) => {
169   const updatedProduct = {
170     name: req.body.name,
171     description: req.body.description,
172     price: Decimal128.fromString(req.body.price.toString()), // store this as 128bit decimal in MongoDB
173     image: req.body.image
174   };
175   MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=
176   .then(client => {
177     client
178     .db()
179     .collection('product')
180     .updateOne({ _id: new ObjectId(req.params.id) }, {
181       $set : updatedProduct
182     })
183     .then(result =>{
184       res.status(200).json({ message: 'Product updated', productId: req.params.id });
185     })
186     .catch(err => {
187       console.log(err);
188       res.status(500)
189       .json({ message : 'An Error Occurred' });
190     });
191   })
192   .catch(err => {
193     res.status(500)
194     .json({ message : 'An Error Occurred' });
195   });
196   // res.status(200).json({ message: 'Product updated', productId: 'DUMMY' });
197 });
198
```

Lalu jangan lupa untuk merestart npm servernya. Selanjutnya coba kita edit salah satu data product misalkan seperti ini :

Title	Tas Hitam
Price	100000
Image URL	http://localhost:3100/images/product-backpack.jpg
Description	Tas cantik babastudio punya
Update Product	

Maka hasilnya akan muncul seperti ini :



I. Cara menghapus product

Untuk menghapus data product coba kita tambahkan script ini :

```

199 // Delete a product
200 // Requires logged in user
201 router.delete('/:id', (req, res, next) => {
202   MongoClient.connect('mongodb+srv://mongodb:b4b4studio2019@cluster0-9wm0a.mongodb.net/shop?retryWrites=
203   .then(client => {
204     client
205     .db()
206     .collection('product')
207     .deleteOne({ _id: new ObjectId(req.params.id) })
208     .then(result => {
209       res.status(200).json({ message: 'Product deleted' });
210     })
211     .catch(err => {
212       console.log(err);
213       res.status(500)
214       .json({ message : 'An Error Occurred' });
215     });
216   })
217   .catch(err => {
218     res.status(500)
219     .json({ message : 'An Error Occurred' });
220   });
221   // res.status(200).json({ message: 'Product deleted' });
222 });
223

```

Lalu kita restart kembali npm server. Lalu kita coba hapus salah 1 product maka datanya akan ke hapus tetapi productnya tidak ke refresh secara otomatis. Untuk itu coba kita buka files products.js di folder src/pages/Product.

```

6 class ProductsPage extends Component {
7   state = { isLoading: true, products: [] };
8   componentDidMount() {
9     axios
10      .get('http://localhost:3100/products')
11      .then(productsResponse => {
12        this.setState({ isLoading: false, products: productsResponse.data });
13      })
14      .catch(err => {
15        this.setState({ isLoading: false, products: [] });
16        this.props.onError('Loading products failed. Please try again later');
17        console.log(err);
18      });
19   }

```

Pada baris ke 9 – 18 kita cut lalu kita buat seperti ini di bawah productdelethandler

```

26 fetchData = () => {
27   axios
28     .get('http://localhost:3100/products')
29     .then(productsResponse => {
30       this.setState({ isLoading: false, products: productsResponse.data });
31     })
32     .catch(err => {
33       this.setState({ isLoading: false, products: [] });
34       this.props.onError('Loading products failed. Please try again later');
35       console.log(err);
36     });
37 }

```

Kita tambahkan lagi di baris ke 9 & 17

```

6 class ProductsPage extends Component {
7   state = { isLoading: true, products: [] };
8   componentDidMount() {
9     this.fetchData();
10  }
11
12  productDeleteHandler = productId => {
13    axios
14      .delete('http://localhost:3100/products/' + productId)
15      .then(result => {
16        console.log(result);
17        this.fetchData();
18      })
19      .catch(err => {
20        this.props.onError(
21          'Deleting the product failed. Please try again later'
22        );
23        console.log(err);
24      });
25  };

```

Lalu restart kembali npm servernya dan coba delete salah 1 productnya lagi.