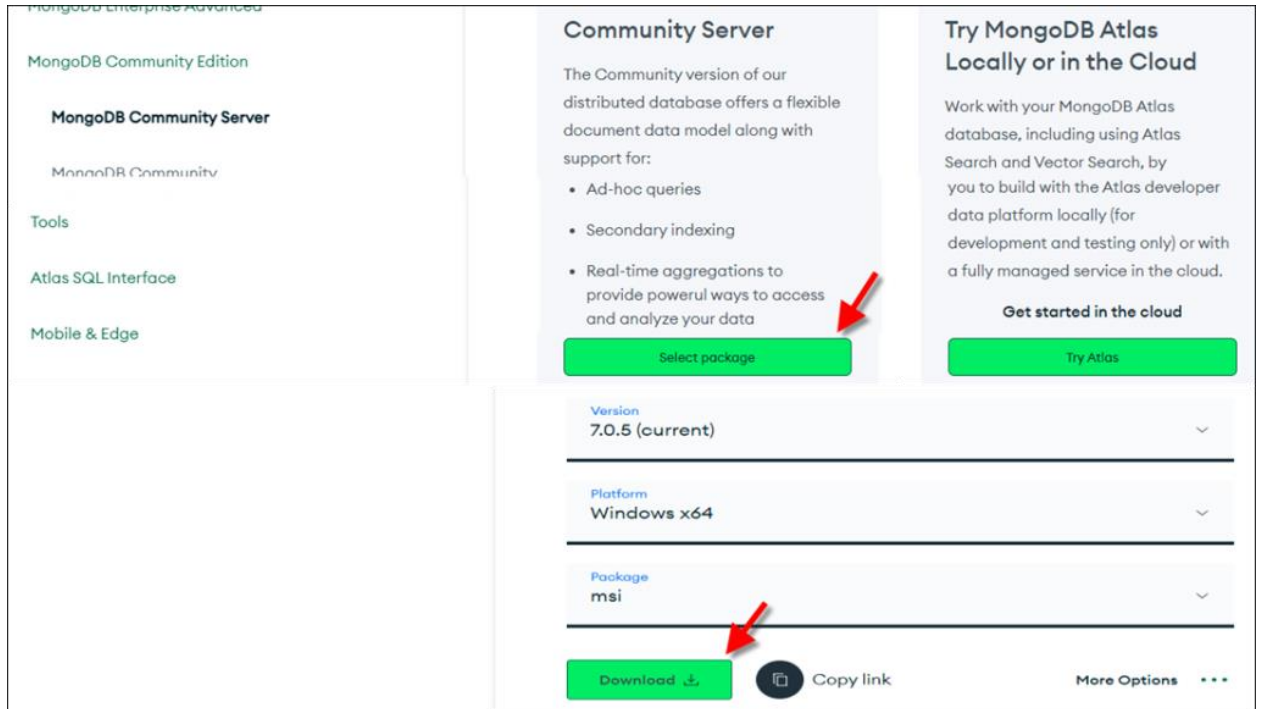


LAB03-LÀM VIỆC VỚI MONGODB TRONG NODEJS

I. Download và cài đặt mongodb

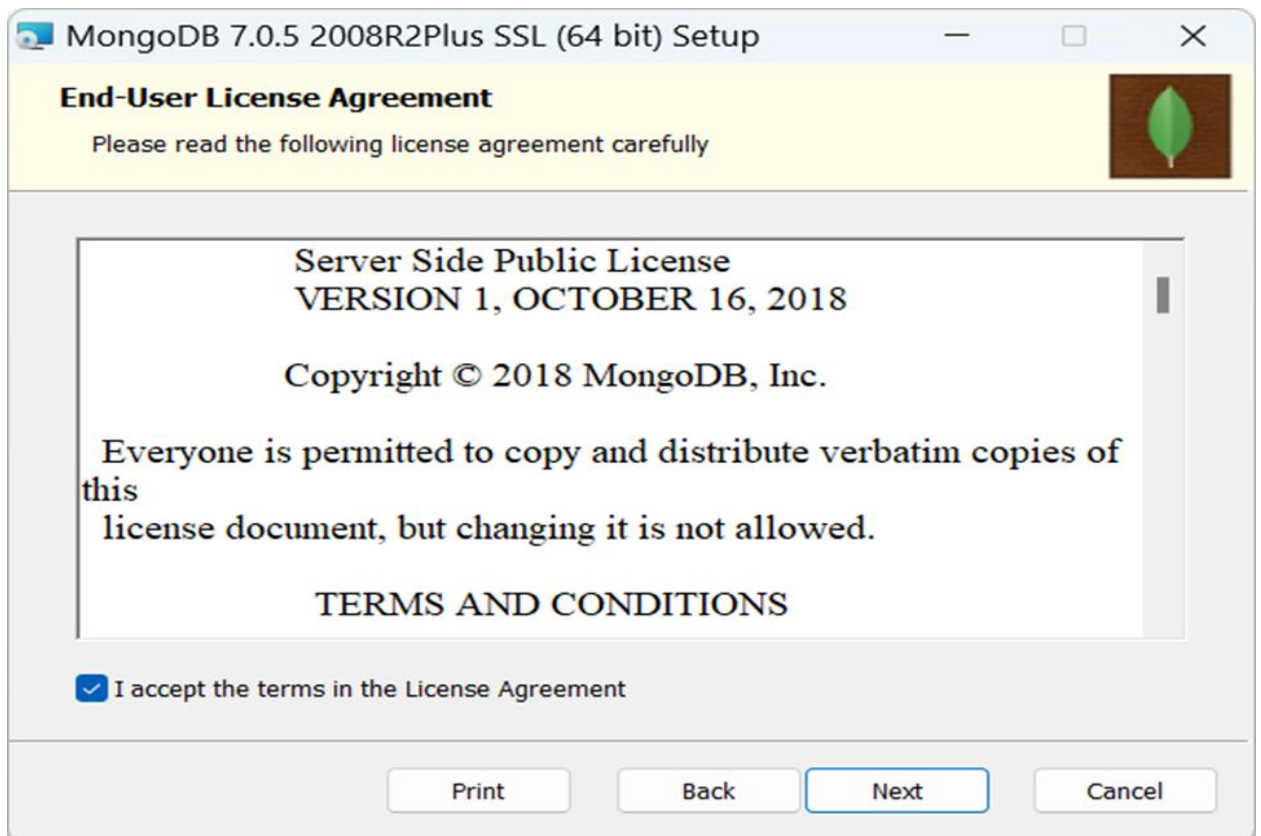
– Vào <https://www.mongodb.com/try/download/community> và download



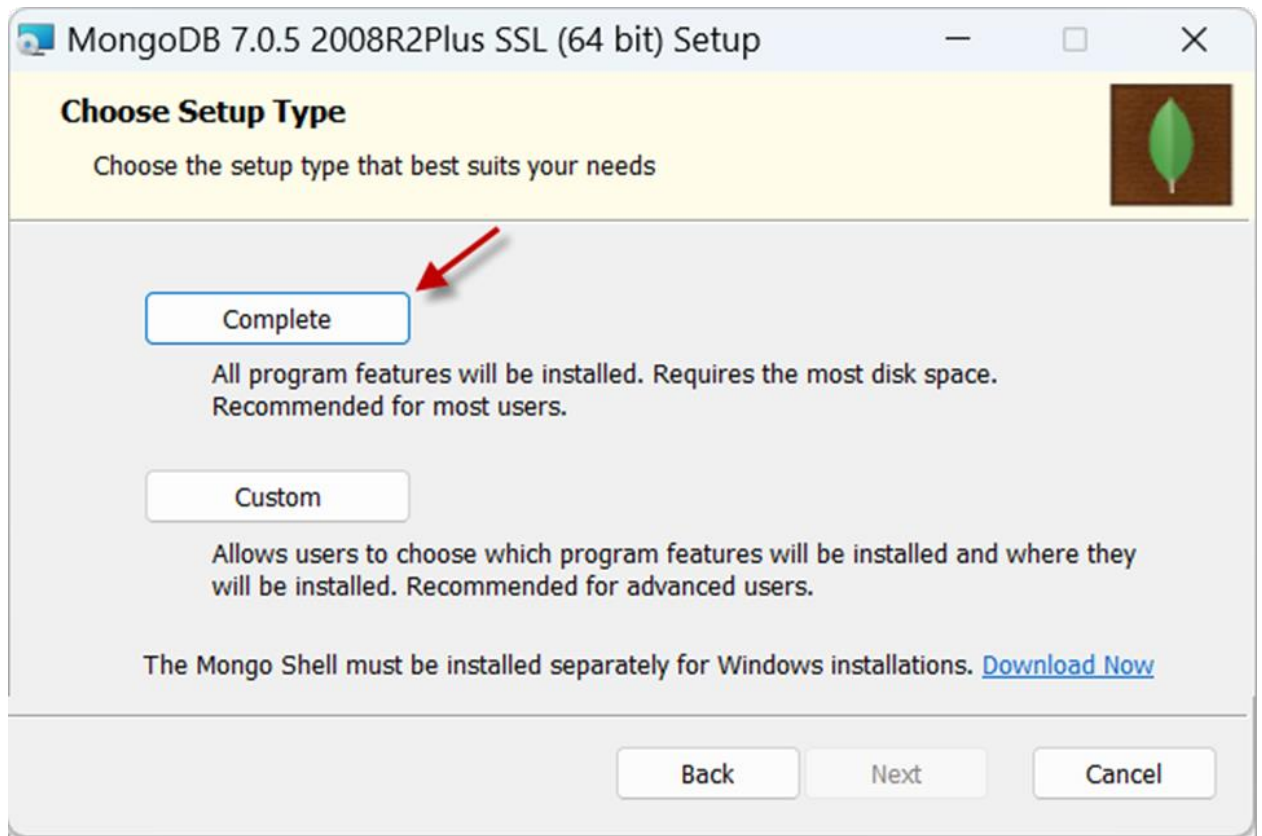
– Chạy file vừa cài đặt rồi nhấn **Next** trong hình để tiếp tục



– Chọn **I accept the terms ...** rồi nhấn Next



– Chọn **Complete** rồi nhấn **Next**



– Chọn **Run service...** để chạy mongo như là service trong máy rồi nhấn **Next**

MongoDB 7.0.5 2008R2Plus SSL (64 bit) Service Cust...

Service Configuration

Specify optional settings to configure MongoDB as a service.

☒ Install MongoDB as a Service

☒ Run service as Network Service user

☐ Run service as a local or domain user:

Account Domain:

Account Name:

Account Password:

Service Name:

Data Directory:

Log Directory:

< Back Next > Cancel

– Chọn **Install MongoDB Compass** – công cụ trực quan quản lý MongoDB rồi nhấn **Next**

MongoDB Compass

Install MongoDB Compass

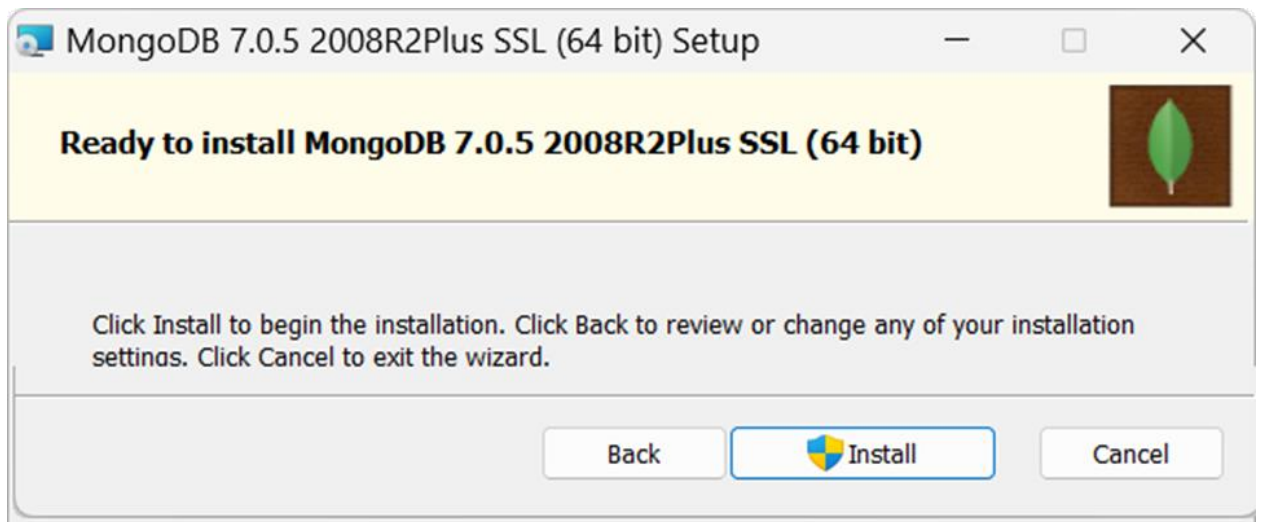
MongoDB Compass is the official graphical user interface for MongoDB.

By checking below this installer will automatically download and install the latest version of MongoDB Compass on this machine. You can learn more about MongoDB Compass here: <https://www.mongodb.com/products/compass>

☒ Install MongoDB Compass

Back Next Cancel

– Nhấp **Install** để cài đặt



– Nhấn **Finish** để kết thúc

II. Các khái niệm trong mongodb

1. Database trong mongodb

Mỗi database trong mongodb giống database trong Mysql. Một database có 1 tên và là kho chứa nhiều Collection.

2. Collection trong mongodb

Collection trong mongodb giống như table trong Mysql. Mỗi collection có 1 tên và đặt trong 1 database nào đó. Khi tạo collection, không cần phải khai báo các cột.

3. Document trong mongodb

Document là các record trong collection. Mỗi document gồm nhiều field , được nhập theo cú pháp json.

4. Field trong document

Field là từng cặp name:value trong document. Một document có nhiều field, cách nhau bởi dấu phẩy. Với các document trong 1 collection thì số field có thể khác nhau.

```
{ "idLoai": 1, "tenLoai": "Nghệ thuật sống", "thuTu": 1, "anhien": false }
```

```
{ "idLoai": 10, "tenLoai": "Thám hiểm", "anhien": true }
```

5. Các kiểu dữ liệu trong MongoDB

MongoDB hỗ trợ các kiểu dữ liệu sau: string, double, int, date, boolean, ObjectID, array...

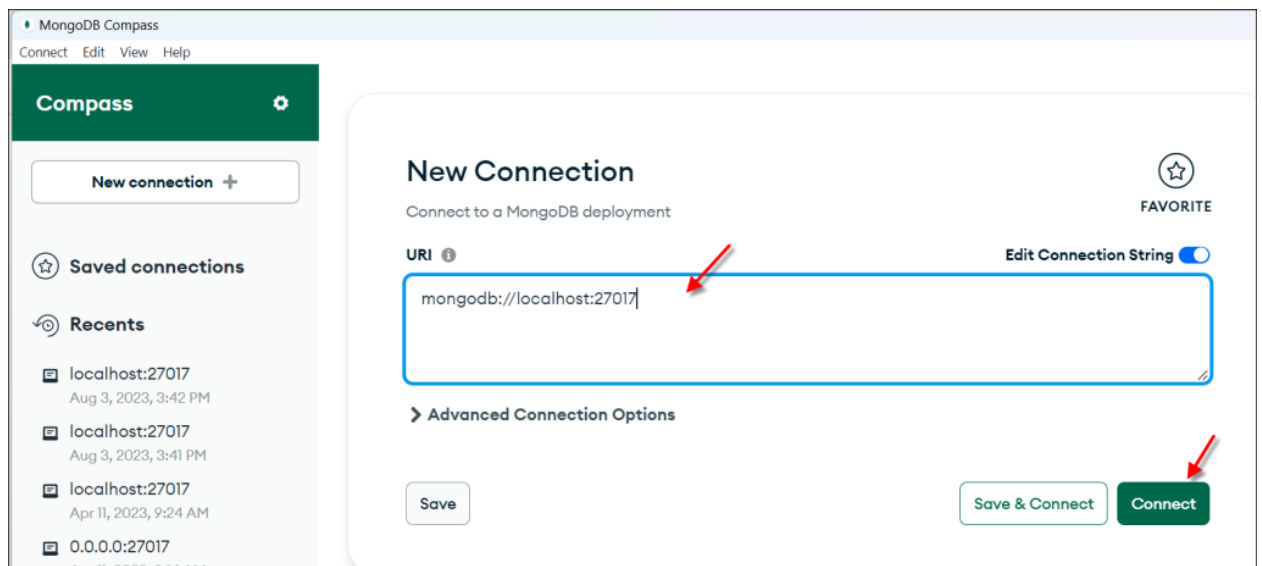
Các công cụ quản trị MongoDB

- **MongodbCompass:** là công cụ quản lý hệ CSDL MongoDB trên desktop, công cụ này được cài kèm theo khi bạn cài đặt MongoDB vào máy tính.
- **Robo 3T:** Robo 3T cũng là công cụ trực quan để làm việc với MongoDB. Giống như dùng phpmyadmin để làm việc với MySQL vậy. Để download Robo 3T, vào trang <https://robomongo.org/download>
- Ngoài ra còn có các công cụ như **Umongo, MongoExplorer, RockMongo...**

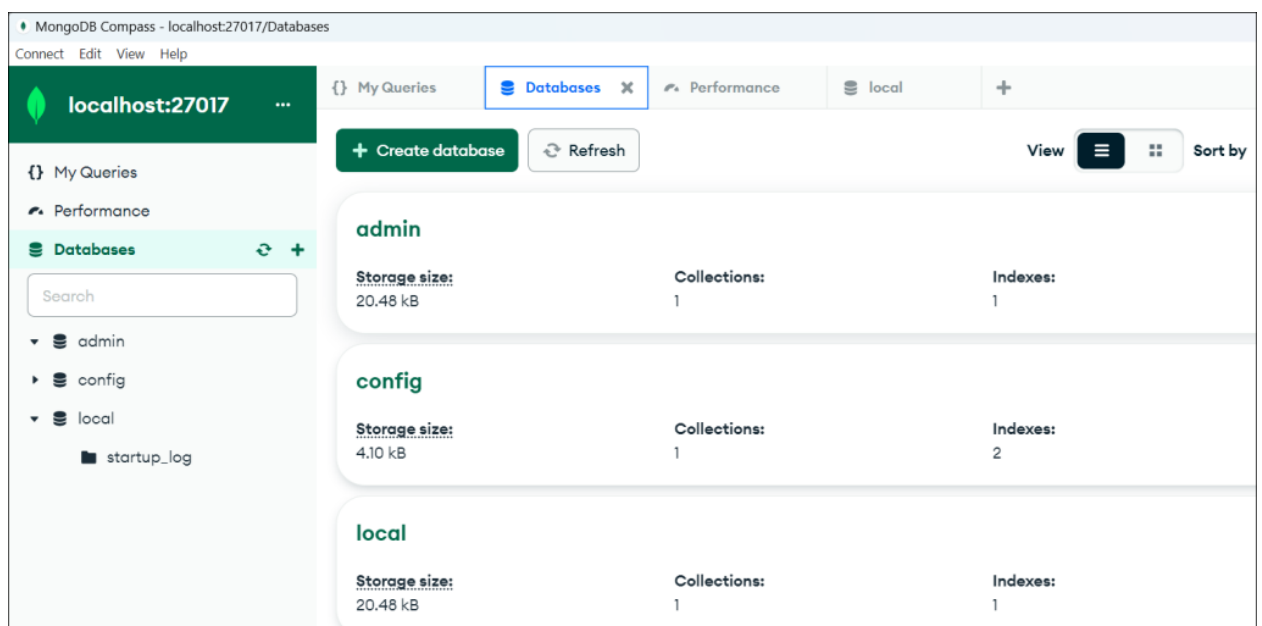
III. Quản trị database mongodb với MongoDBCompass

1. Kết nối mongodb trên máy local

– Mở MongoDB Compass => nhấp Connect

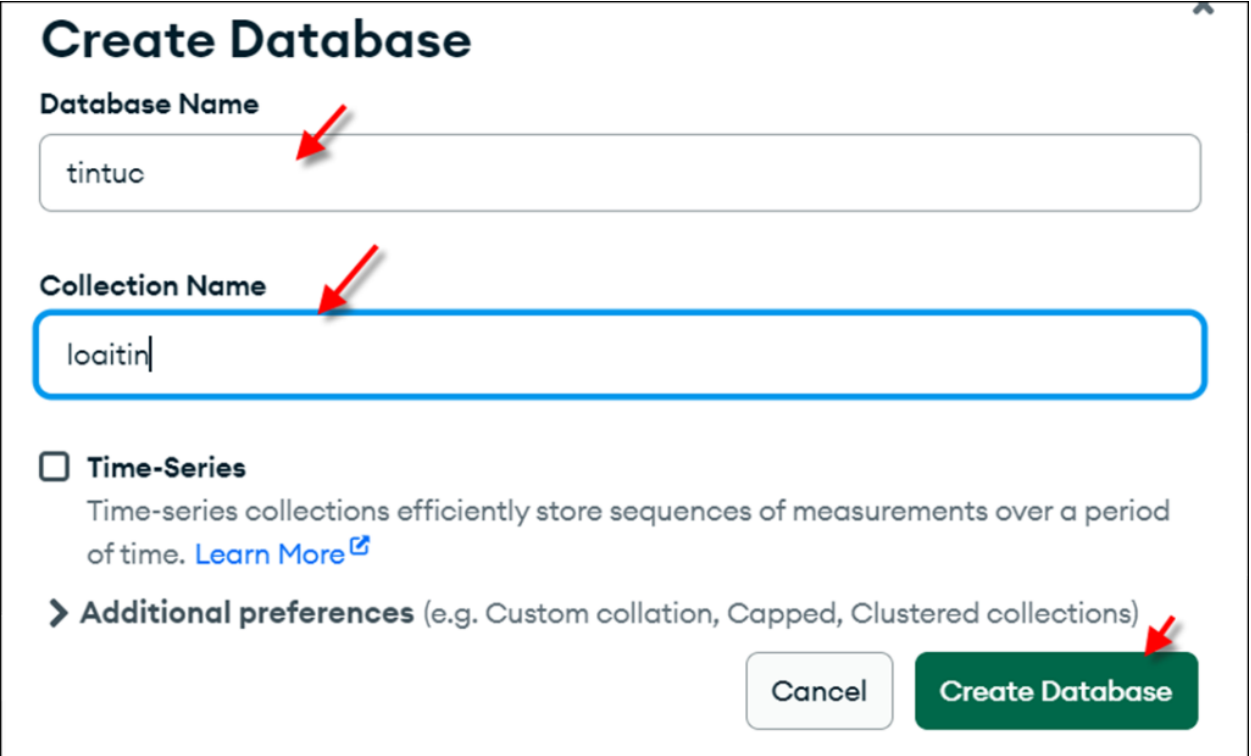


– Khi kết nối thành công, cột trái sẽ hiện thông tin kết nối như sau:



2. Tạo database mongodb

Nhấp nút Create Database (xem hình trên) rồi nhập tên Database + nhập tên collection đầu tiên trong database rồi nhấn nút Create Database



Create Database

Database Name

tintuc

Collection Name

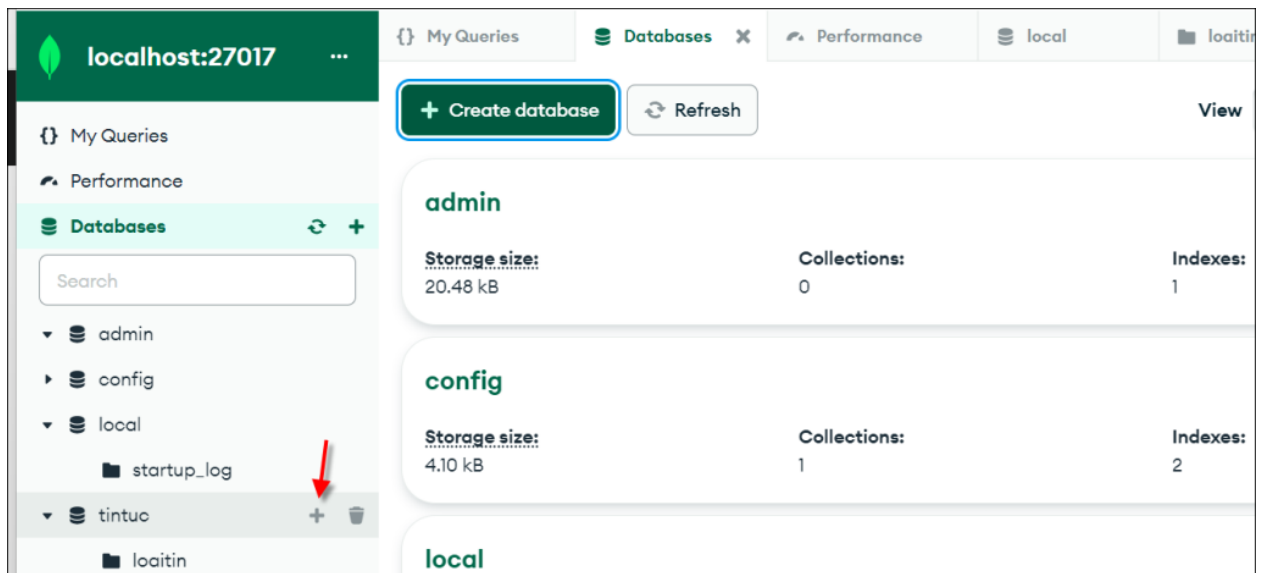
loaitin

☐ **Time-Series**
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

➤ **Additional preferences** (e.g. Custom collation, Capped, Clustered collections)

Cancel Create Database


Kết quả tạo:



3. Tạo collection (table)

Nhấp nút + ở phía sau tên database (xem hình trên) rồi nhập tên collection , rồi nhấn nút Create Collection

Create Collection


Collection Name 

tin|

☐ Time-Series
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Capped, Clustered collections)

Cancel

Create Collection 

4. Chèn document

Mỗi document là 1 record dữ liệu cần lưu. Bạn nhập tên collection rồi nhấn **Add Data** ==> Insert Document

localhost:27017

My QueriesPerformanceDatabaseslocalloaitin

My QueriesPerformanceDatabases

Search

adminconfiglocaltintucloaitintin


tintuc.loaitin

DocumentsAggregationsSchemaIndexesValidation

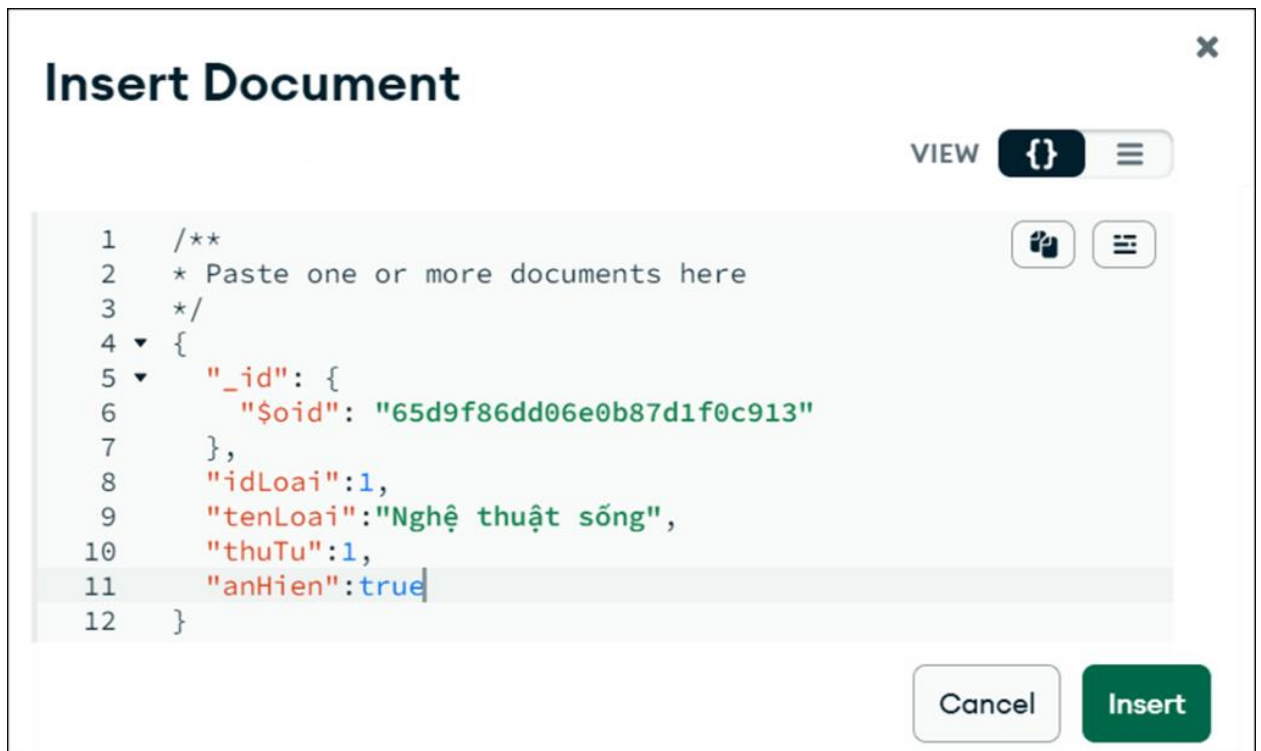
FilterType a query: { field: 'value' } or Generate query

ADD DATAEXPORT DATAUPDATEDELETE

Import JSON or CSV file

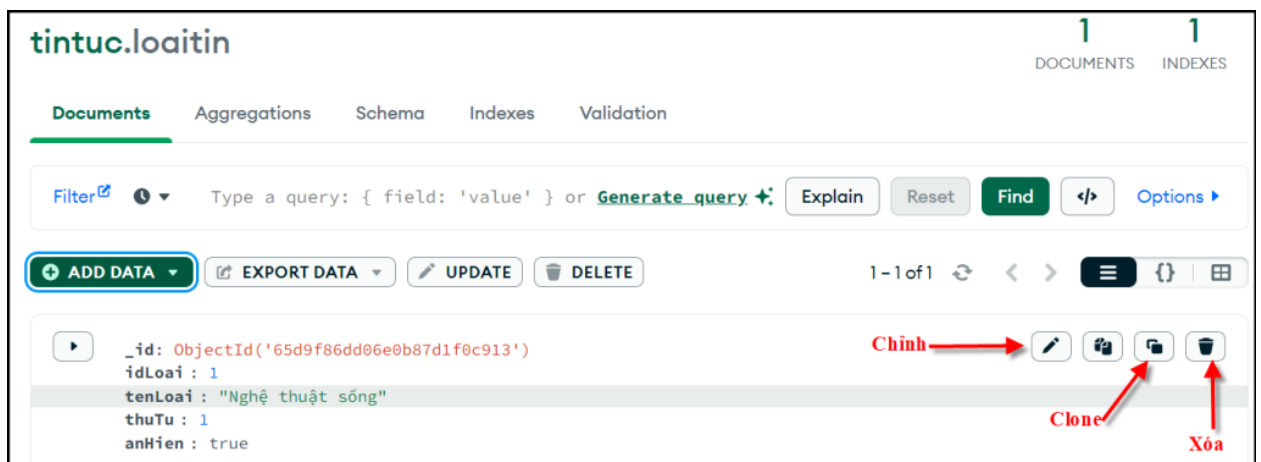
Insert document 

Rồi nhập dữ liệu vào , xong nhấn nút **Insert**



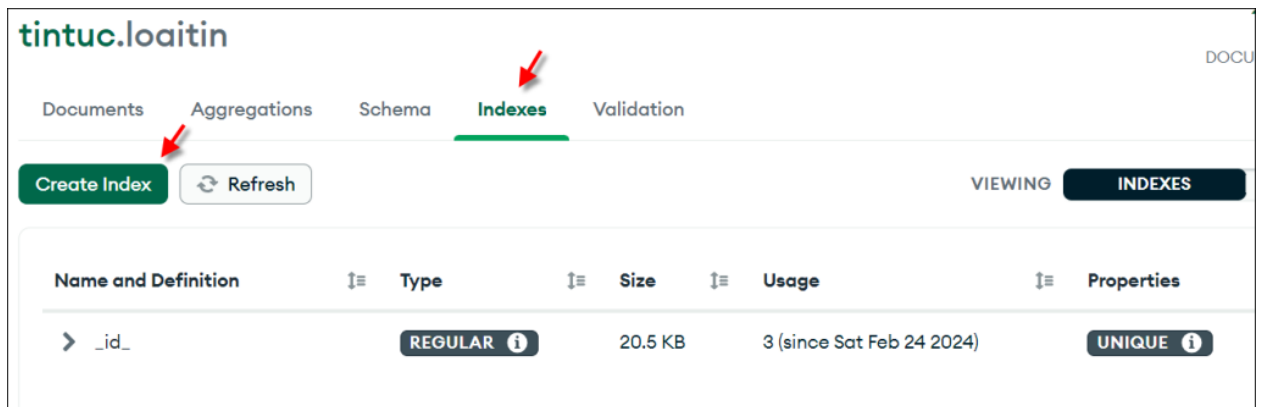
5. Clone document , cập nhật, xóa document

Nhấp các nút tương ứng như trong hình dưới



6. Tạo index cho documents trong collection

Tạo index để sắp xếp sẵn các record nhằm phục vụ tìm kiếm cho nhanh chóng. Thực hiện bằng cách nhấp tab **Indexes** rồi nhấp nút **Create Index**

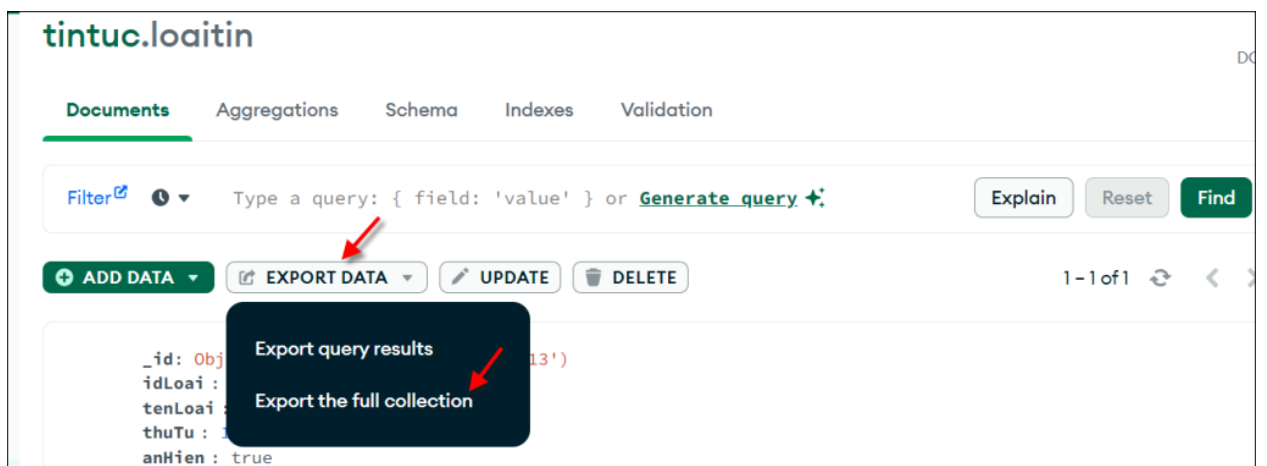


Sau đó chọn field , kiểu Index và nhấn nút **Create index**



7. Export documents trong collection

– Nhấn nút **Export** như hình dưới



– Chọn **JSON** rồi nhấn nút **Export**

Export



Collection tintuc.loaitin

Export File Type

JSON

CSV

> Advanced JSON Format

Cancel

Export...

IV. Một số hàm NodeJS dùng để tương tác với MongoDB

1. Chèn document (insert record)

Dùng hàm insertOne

```
db.people.insertOne( { user_id: "bcd001", age: 45, status: "A" } )  
// INSERT INTO people(user_id, age, status) VALUES ("bcd001", 45, "A")
```

2. Chọn document (select records)

a. Chọn tất cả các document trong collection

Dùng hàm find như sau

```
db.people.find()  
~ SELECT * FROM people
```

b. Chọn các document trong collection theo điều kiện

Dùng hàm find với tham số là điều kiện chỉ ra

```
db.people.find( { status: "A" } )  
~ SELECT * FROM people WHERE status = "A"  
  
db.people.find( { status: "A", age: 50 } )  
~ SELECT * FROM people WHERE status = "A" AND age = 50  
  
db.people.find( { $or: [ { status: "A" } , { age: 50 } ] } )  
~ SELECT * FROM people WHERE status = "A" OR age = 50  
  
db.people.find( { user_id: /bc/ } )  
  
db.people.find( { user_id: { $regex: /bc/ } } ) ~ SELECT *  
FROM people WHERE user_id like "%bc%"
```

3. Đếm các document trong collection

Dùng hàm count để đếm

```
db.people.count( { user_id: { $exists: true } } )  
  
db.people.find( { user_id: { $exists: true } } ).count()  
~ SELECT COUNT(user_id) FROM people
```

4. Cập nhật document trong collection

Dùng hàm `updateMany` để cập nhật nhiều, `updateOne` để cập nhật 1 document

```
db.people.updateMany( { age: { $gt: 25 } }, { $set: { status: "C" } } )
```

```
~ UPDATE people SET status = "C" WHERE age > 25
```

```
db.people.updateMany( { status: "A" } , { $inc: { age: 3 } } )
```

```
~ UPDATE people SET age = age + 3 WHERE status = "A"
```

5. Xóa document trong collection

Dùng hàm `deleteMany` để xóa nhiều, hàm `deleteOne` để xóa 1 document.

```
db.people.deleteMany( { status: "D" } )
```

```
~ DELETE FROM people WHERE status = "D"
```

```
db.people.deleteMany({})
```

```
~ DELETE FROM people
```

TƯƠNG TÁC MONGODB TỪ NODEJS

MỤC TIÊU: Sinh viên thực hành áp dụng mongodb, nodejs để chọn dữ liệu, thêm, sửa, xóa dữ liệu trong mongodb từ trong project NodeJS.

1. Tạo project NodeJS và cài module

Chạy lệnh tạo project:

```
express --ejs Node_Mongodb
```

Cài đặt các module vào project: Mở command line rồi chuyển vào folder project gõ lệnh:

```
npm install  
npm install mongodb
```

2. Nhúng hàm MongoClient trong module mongodb vào

```
//index.js  
const { MongoClient } = require("mongodb");
```

3. Chèn document vào mongodb từ NodeJS:

```
//routes/index.js  
router.get("/chenRecord", async (req, res) => {  
  const uri = "mongodb://localhost:27017";  
  const client = new MongoClient(uri);  
  await client.connect();  
  console.log('Kết nối thành công đến server');  
  
  const db = client.db('tintuc');  
  let doc1 = { "idLoai":10, "tenLoai": "Khoa học","thuTu":  
10, "anhien":false };  
  const loaitin = db.collection('loaitin');  
  const insertResult = await loaitin.insertOne(doc1);  
  res.status(200).send('Đã chèn xong . InsertedID= ' +  
insertResult.insertedId);  
  client.close();  
});
```

Test: xem trong trình duyệt <http://localhost:3000/chenRecord> sẽ thấy thông báo chèn thành công.

← → ↻ ⓘ localhost:3000/chenRecord					
Đã chèn xong . InsertedID= 65dd75ec5e171832d945bbcd					
<div> ➕ ADD DATA 📄 EXPORT DATA ✎ UPDATE 🗑 DELETE </div> <div>1 - 6 of 6</div>					
🏠 loaitin					
	_id ObjectId	idLoai Int32	tenLoai String	thuTu Int32	anHien
1	ObjectId('65d9f86dd06e0b8...	1	"Nghệ thuật sống"	1	true
2	ObjectId('65dd68fc3a7c7fd...	2	"Thể thao"	2	true
3	ObjectId('65dd690b3a7c7fd...	3	"Sức khỏe"	3	true
4	ObjectId('65dd69263a7c7fd...	4	"Ẩm thực"	4	true
5	ObjectId('65dd693a3a7c7fd...	5	"Du lịch"	5	false
6	ObjectId('65dd75ec5e17183...	10	"Khoa học"	10	No file

Tương tự, các em chèn loại tin: Nghệ thuật sống, thể thao, Sức khỏe, Ẩm thực, Du lịch, Khoa học.

4. Cập nhật document trong mongodb từ NodeJS

```
//routes/index.js
router.get("/capnhat", async (req, res) => {
  var uri = "mongodb://localhost:27017";
  const client = new MongoClient(uri);
  await client.connect();
  console.log('Kết nối thành công đến server');

  const db = client.db('tintuc');
  const loaitin = db.collection('loaitin');
  let myquery = { thuTu: 10 };
  let values = { $set: {"tenLoai": 'Đời sống', "thuTu": 15 }
};
  const kq = await loaitin.updateOne(myquery, values);
  console.log("kq =", kq)
  res.status(200).send('Đã cập nhật xong ' + kq.
matchedCount + ' dòng' );
  client.close();
});
```

Test: xem trong trình duyệt <http://localhost:3000/capnhat> sẽ thấy thông báo cập nhật thành công.

← → ↻ ⓘ localhost:3000/capnhat					
Đã cập nhật xong 1 dòng					
<div> ADD DATA EXPORT DATA UPDATE DELETE </div> <div>1 - 6 of 6</div>					
🏠 loaitin					
	_id ObjectId	idLoai Int32	tenLoai String	thuTu Int32	anHien
1	ObjectId('65d9f86dd06e0b8...	1	"Nghệ thuật sống"	1	true
2	ObjectId('65dd68fc3a7c7fd...	2	"Thể thao"	2	true
3	ObjectId('65dd690b3a7c7fd...	3	"Sức khỏe"	3	true
4	ObjectId('65dd69263a7c7fd...	4	"Ẩm thực"	4	true
5	ObjectId('65dd693a3a7c7fd...	5	"Du lịch"	5	false
6	ObjectId('65dd75ec5e17183...	10	"Đời sống"	15	No fi

5. Cập nhật nhiều document trong mongodb từ NodeJS

```
//routes/index.js
router.get("/capnhatn", async (req, res) => {
  const uri = "mongodb://localhost:27017";
  const client = new MongoClient(uri);
  await client.connect();
  console.log('Kết nối thành công đến server');

  const db = client.db('tintuc');
  const loaitin = db.collection('loaitin');
  let myquery = { thuTu: { $lt: 5 } };
  let values = { $set: { thuTu: 1, } };
  const kq = await loaitin.updateMany(myquery, values);
  console.log("kq =", kq)
  res.status(200).send('Đã cập nhật xong ' + kq.
  matchedCount + ' dòng' );
  client.close();
});
```

Test: xem trong trình duyệt <http://localhost:3000/capnhatn> sẽ thấy thông báo cập nhật thành công.

localhost:3000/capnhatn					
Đã cập nhật xong 4 dòng					
<div> ADD DATA EXPORT DATA UPDATE DELETE </div> <div>1 - 6 of 6</div>					
loaitin					
	_id ObjectId	idLoai Int32	tenLoai String	thuTu Int32	anHien
1	ObjectId('65d9f86dd06e0b8...	1	"Nghệ thuật sống"	1	true
2	ObjectId('65dd68fc3a7c7fd...	2	"Thể thao"	1	true
3	ObjectId('65dd690b3a7c7fd...	3	"Sức khỏe"	1	true
4	ObjectId('65dd69263a7c7fd...	4	"Ấm thực"	1	true
5	ObjectId('65dd693a3a7c7fd...	5	"Du lịch"	5	true
6	ObjectId('65dd75ec5e17183...	10	"Đời sống"	15	No fi

6. Xóa document trong mongodb từ NodeJS

```
//routes/index.js
router.get("/xoaRecord", async (req, res) => {
  const uri = "mongodb://localhost:27017";
  const client = new MongoClient(uri);
  await client.connect();
  console.log('Kết nối thành công đến server');

  const db = client.db('tintuc');
  const loaitin = db.collection('loaitin');
  let myquery = {"thuTu": 15 } ;
  const kq = await loaitin.deleteOne(myquery);
  res.status(200).send('Đã xóa xong ' + kq.deletedCount + ' dòng' );
  client.close();
});
```

Test: xem trong trình duyệt <http://localhost:3000/xoaRecord> sẽ thấy thông báo xóa thành công.

localhost:3000/xoaRecord					
Đã xóa xong 1 dòng					
<div> ADD DATA EXPORT DATA UPDATE DELETE </div> <div>1 - 5 of 5</div>					
loaitin					
	_id ObjectId	idLoai Int32	tenLoai String	thuTu Int32	anHien
1	ObjectId('65d9f86dd06e0b8...	1	"Nghệ thuật sống"	1	true
2	ObjectId('65dd68fc3a7c7fd...	2	"Thể thao"	1	true
3	ObjectId('65dd690b3a7c7fd...	3	"Sức khỏe"	1	true
4	ObjectId('65dd69263a7c7fd...	4	"Ấm thực"	1	true
5	ObjectId('65dd693a3a7c7fd...	5	"Du lịch"	5	true

7. Select document trong mongodb và hiện kết quả trong NodeJS

```
//routes/index.js
router.get("/layRecord", async (req, res) => {
  let uri = "mongodb://localhost:27017";
  const client = new MongoClient(uri);
  await client.connect();
  console.log('Kết nối thành công đến server');

  const db = client.db('tintuc');
  const loaitin = db.collection('loaitin');
  let myquery = {} ;
  const arr = await loaitin.find(myquery).toArray();
  client.close();
  res.render("loaitin",{listloaitin:arr});
});
```

Tạo file views/loaitin.ejs

```
<h1>Danh sách loại tin</h1>
<% for (let lt of listloaitin) {%>
  <p> <%=lt.tenLoai%></p>
<% } %>
```

Test: xem trong trình duyệt <http://localhost:3000/layRecord> sẽ thấy dữ liệu hiện ra



8. Select 1 document trong mongodb từ NodeJS

```
//routes/index.js
router.get("/layRecord/:id", async (req, res) => {
  let id= parseInt(req.params.id);
  let uri = "mongodb://localhost:27017";
  const client = new MongoClient(uri);
  await client.connect();
  console.log('Kết nối thành công đến server');

  const db = client.db('tintuc');
  const loaitin = db.collection('loaitin');
  let myquery = { idLoai:id };
  let data = await loaitin.findOne(myquery);
  res.render("chitietloai",{loaitin:data});
  client.close();
})
```

Test: xem trong trình duyệt <http://localhost:3000/layRecord/1> sẽ thấy dữ liệu hiện ra



BÀI TẬP:

1. Trang hiện danh sách loại tin : định dạng cho đẹp
2. Tạo form thêm loại tin để thêm loại tin mới vào collection loaitin
3. Tạo form cập nhật 1 loại tin vào collection loaitin
4. Tạo chức năng xóa 1 loại tin trong collection loaitin
5. Tạo form thêm tin để thêm tin mới vào collection tin
6. Tạo trang hiện danh sách các tin từ collection tin