

# First MEAN partie 2

## Objectif

Créer un projet avec la stack MEAN  
Créer un serveur avec node et express  
Créer un front end angular  
gérer le cross origin resource sharing

## Get et Post

Modifions le middleware generique en get et post

```
18
19 app.post("/api/posts", (req, res, next) => {
20   const post = req.body;
21   console.log(post);
22   res.status(201).json({
23     message: "Post added",
24   });
25   next();
26 });
27
28 // middleware
29 app.get("/api/posts", (req, res, next) => {
30   const posts = [
31     { id: "id", title: "title", content: "form express" },
32     { id: "id2", title: "title2", content: "form express2" },
33   ];
34
35   res.status(200).json({
36     message: "Posts from server",
37     posts: posts,
38   });
39 });
40
41 module.exports = app;
```

Dans post.create.ts

```

5
6 @Component({
7   selector: "app-post-create",
8   templateUrl: "./post-create.component.html",
9   styleUrls: ["./post-create.component.css"]
10 })
11 export class PostCreateComponent {
12   enteredTitle = "";
13   enteredContent = "";
14
15   constructor(public postsService: PostsService) {}
16
17   onAddPost(form: NgForm) {
18     if (form.invalid) {
19       return;
20     }
21     console.log(form.value.title);
22
23     this.postsService.addPost(form.value.title, form.value.content);
24     form.resetForm();
25   }
26 }

```

tester

MyMessages

Post Title

Post Content

Save Post

title

title2

Elements

top

Default levels

No Issues

on init

Angular is running in development mode.

service ▶ (2) [{...}, {...}]

les posts liste :

▼ (2) [{...}, {...}] ⓘ

▶ 0: {id: 'id', title: 'title', content: 'content'}

▶ 1: {id: 'id2', title: 'title2', content: 'content2'}

length: 2

▶ [[Prototype]]: Object

lklkjl

test post

>

Il nous faut ajouter le modele

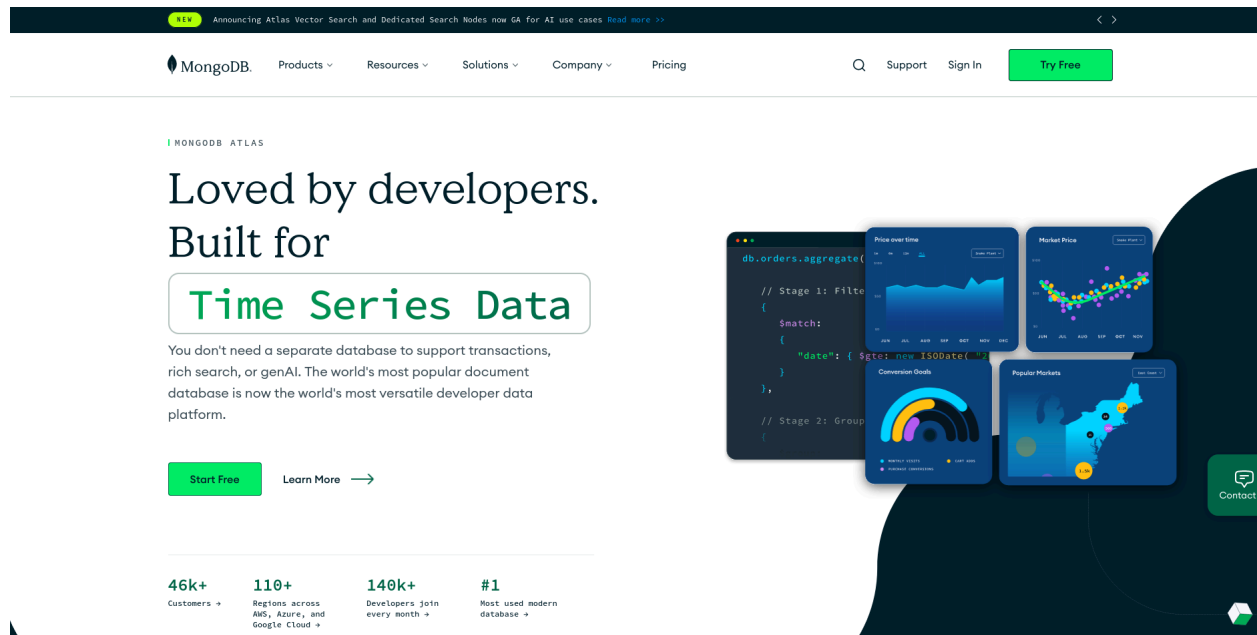
Pour extraire facilement la requête, il a un package nommé body-parser qui n'est plus nécessaire depuis la 4.16

```
backend > app.js > ...  
1  const express = require("express");  
2  
3  
4  const app = express();  
5  app.use(express.json());  
6
```

# Monogodb et Mongoose

Ajoutons une base de données

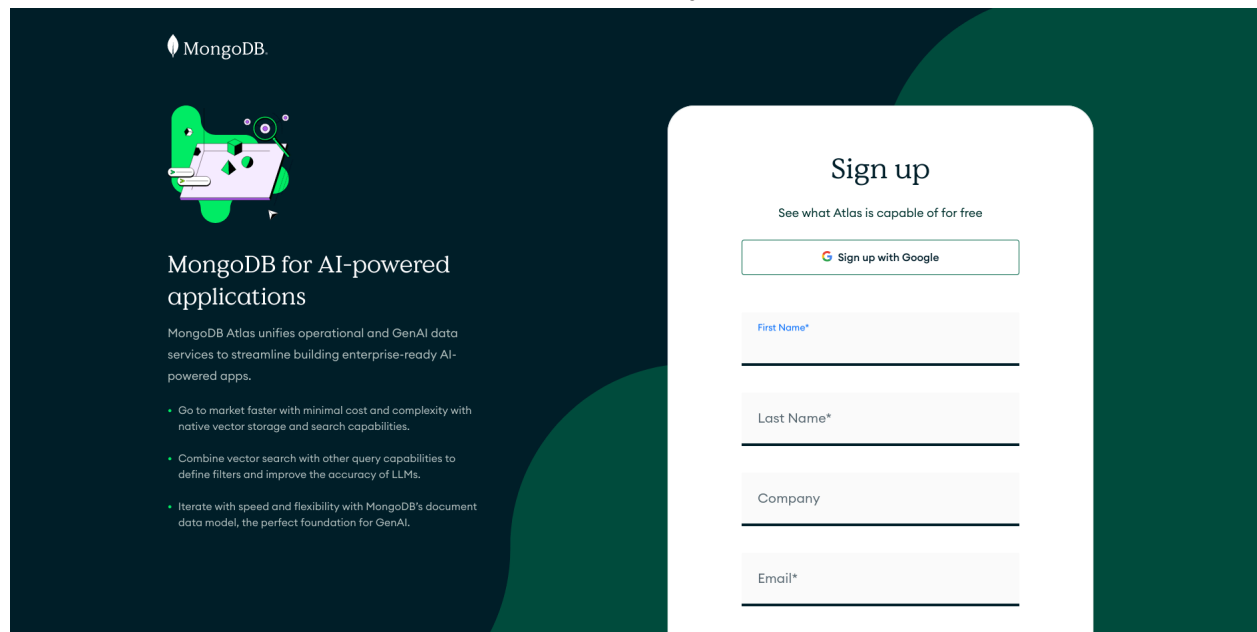
Aller sur <https://www.mongodb.com/>



The screenshot shows the MongoDB Atlas homepage. At the top, there's a dark navigation bar with the MongoDB logo and links for Products, Resources, Solutions, Company, and Pricing. A search bar and a 'Try Free' button are also present. Below the navigation bar, the main heading reads 'Loved by developers. Built for Time Series Data'. A subheading states: 'You don't need a separate database to support transactions, rich search, or genAI. The world's most popular document database is now the world's most versatile developer data platform.' To the right, there are several interactive cards showing code snippets, time series charts, and market data. At the bottom, there are statistics: 46k+ Customers, 110+ Regions across AWS, Azure, and Google Cloud, 140k+ Developers join every month, and #1 Most used modern database. A 'Start Free' button and a 'Learn More' link are also visible.

Try free !

Créer un compte ou se connecter si vous en avez déjà une



The screenshot shows the MongoDB sign-up page. On the left, there's a section titled 'MongoDB for AI-powered applications' with a subheading 'MongoDB Atlas unifies operational and GenAI data services to streamline building enterprise-ready AI-powered apps.' Below this, there are three bullet points: 'Go to market faster with minimal cost and complexity with native vector storage and search capabilities.', 'Combine vector search with other query capabilities to define filters and improve the accuracy of LLMs.', and 'Iterate with speed and flexibility with MongoDB's document data model, the perfect foundation for GenAI.' On the right, there's a 'Sign up' form with a subheading 'See what Atlas is capable of for free'. The form includes a 'Sign up with Google' button, and input fields for 'First Name\*', 'Last Name\*', 'Company', and 'Email\*'. A 'Contact' button is also visible at the bottom right.



## Accept Privacy Policy & Terms of Service

Please acknowledge the following terms and conditions to finish creating your account.

☐ I accept the [Privacy Policy](#) and the [Terms of Service](#)

Cancel Signup

Submit

répondre au questionnaire



## Welcome to Atlas. Let's build something great.

Help us tailor your experience by taking a minute to answer the questions below.

### GETTING TO KNOW YOU

What is your primary goal?

Select

How long have you been developing software with MongoDB?

Select

### GETTING TO KNOW YOUR PROJECT

What programming language are you primarily building on MongoDB with?

Select

What kind(s) of data will your project use?

You can choose as many as you want

Select

Will your application include any of the following architectural models?

You can choose as many as you want

Select

Finish

choisir free à gauche



## Deploy your database

Use a template below or set up [advanced configuration options](#). You can also edit these configuration options once the cluster is created.

**M10**  
\$0.09/hour  
For production applications with sophisticated workload requirements.

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

**SERVERLESS**  
For application development and testing, or workloads with variable traffic.

STORAGE	RAM	vCPU
Up to 1 TB	Auto-scale	Auto-scale

**M0**  
FREE  
For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM	vCPU
512 MB	Shared	Shared

Provider



Region



★ Recommended ⓘ Low carbon emissions ⓘ

Name

You cannot change the name once the cluster is created.

Cluster0

\$0.09/hour



**Pay-as-you-go!** You will be billed hourly and can terminate your cluster anytime. Excludes variable [data transfer](#), [backup](#), and taxes.

[I'll deploy my database later](#)

**M10**  
\$0.09/hour  
For production applications with sophisticated workload requirements.

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

**SERVERLESS**  
For application development and testing, or workloads with variable traffic.

STORAGE	RAM	vCPU
Up to 1 TB	Auto-scale	Auto-scale

**M0**  
FREE  
For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM	vCPU
512 MB	Shared	Shared

Provider



Region



★ Recommended ⓘ Low carbon emissions ⓘ

Name

You cannot change the name once the cluster is created.

Cluster0

Tag (optional)

Create your first tag to categorize and label your resources; more tags can be added later. [Learn more.](#)

Select or enter key : Select or enter value

FREE

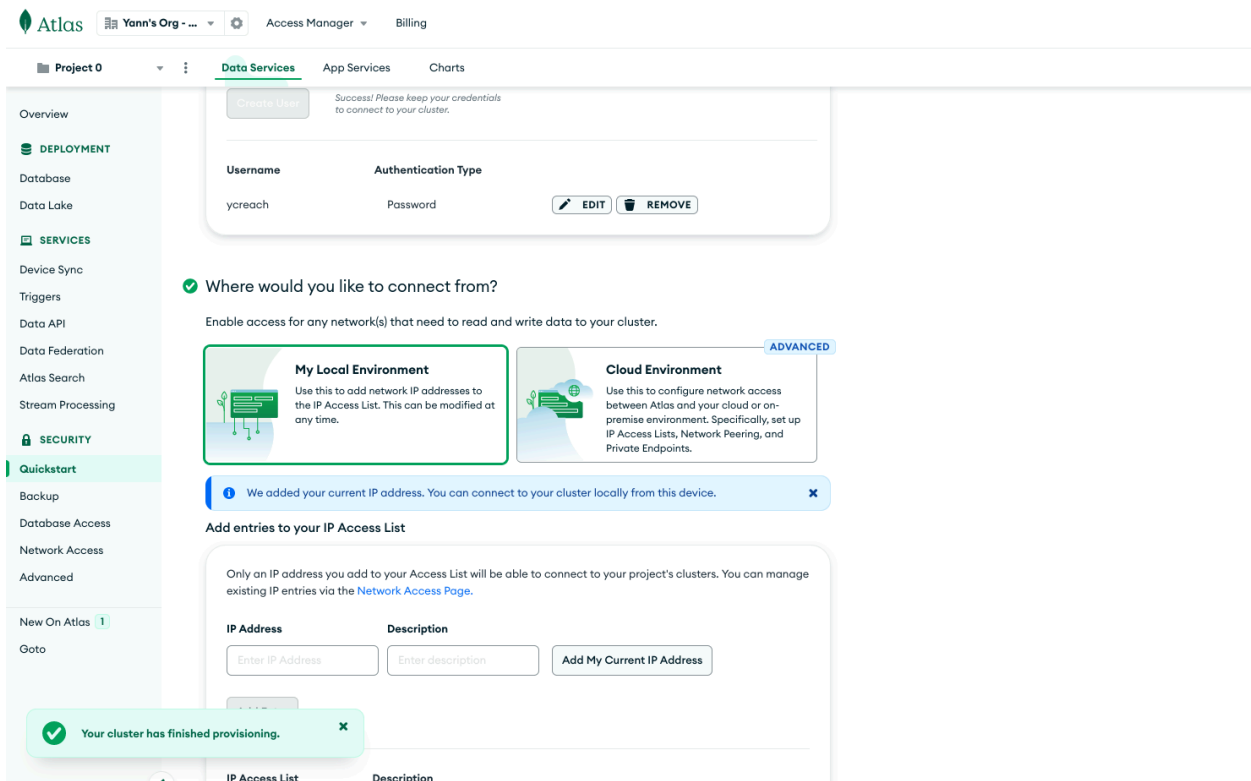


**Free forever!** Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

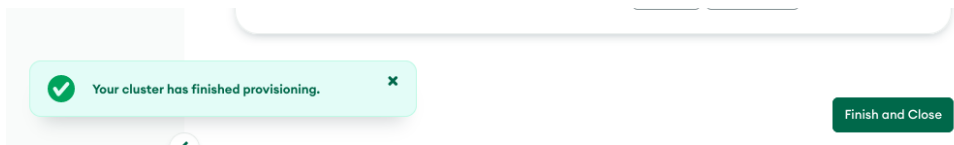
[I'll deploy my database later](#)

create

choisir son type de connection  
password

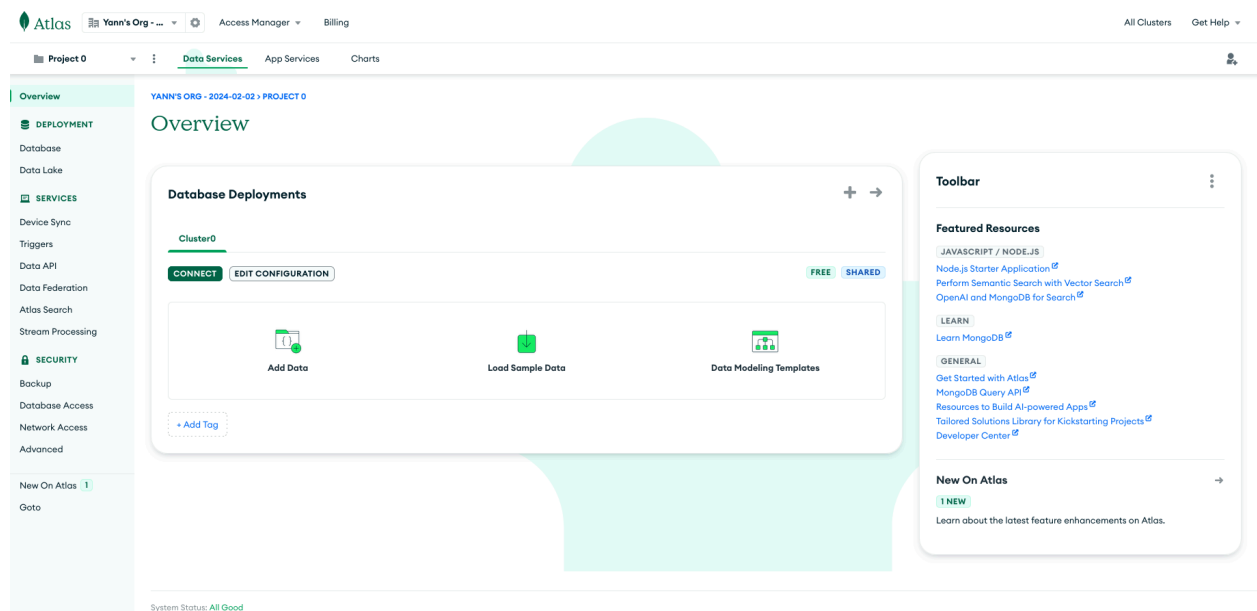


Local environment  
et  
add my current address

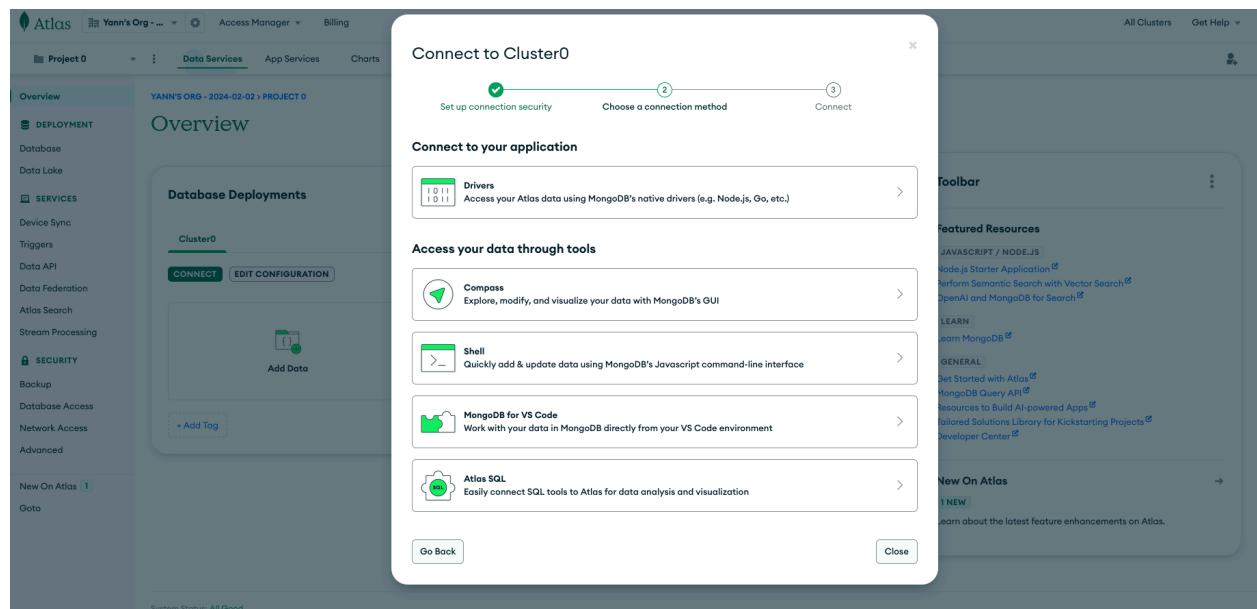


finish

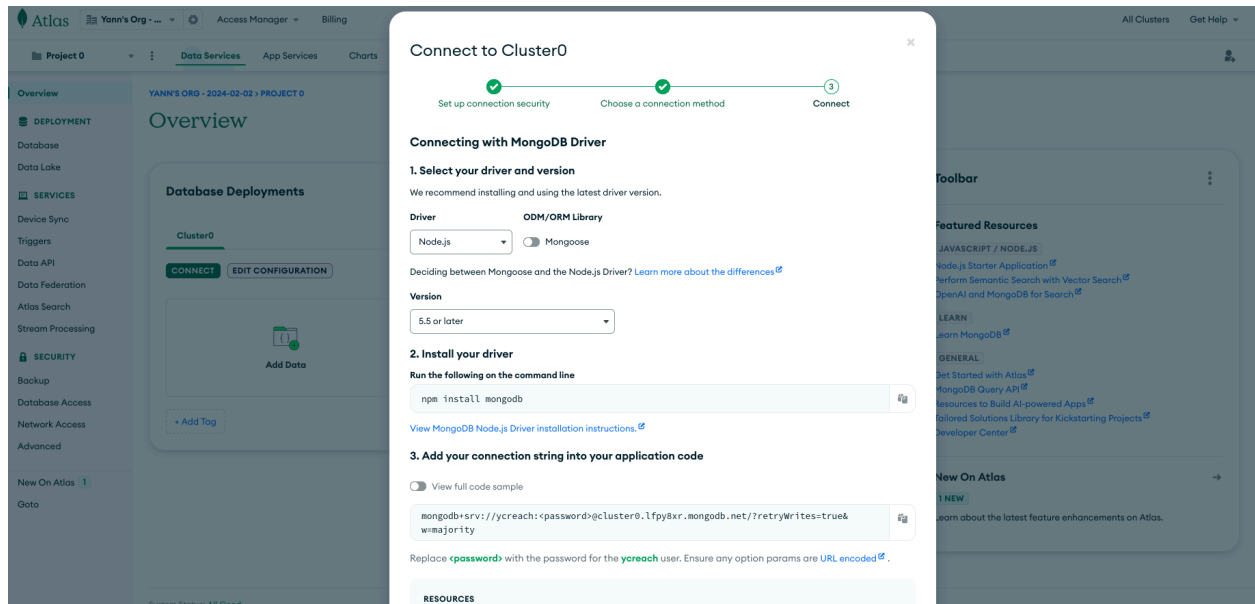




connect



choisir driver



## Conserver cette url



## Mongoose : Un 'Driver' pour angular

Mongoose est une bibliothèque pour Node.js qui fournit une solution de modélisation des objets MongoDB (un ODM - Object Data Modeling) de manière schématique. Elle sert d'interface pour faciliter les interactions entre votre application Node.js et une base de données MongoDB

Mongoose inclut des mécanismes intégrés pour la validation, la requête, les hooks (middlewares), et plus encore, rendant le travail avec MongoDB plus intuitif et sécurisé comparé à l'utilisation du pilote MongoDB natif seul.

Une fois un schéma défini, Mongoose vous permet de créer des modèles basés sur ces schémas. Les modèles sont des constructeurs qui prennent des documents MongoDB et les enveloppent avec des fonctionnalités de modélisation.

# mongoose

elegant **mongodb** object modeling for **node.js**

read the docs

discover plugins

Star 26,367

Version 5.1.0

Fork 3,826

Let's face it, **writing MongoDB validation, casting and business logic boilerplate is a drag**. That's why we wrote Mongoose.

```
const mongoose = require('mongoose');
mongoose.connect('mongodb://127.0.0.1:27017/test');

const Cat = mongoose.model('Cat', { name: String });

const kitty = new Cat({ name: 'Zildjian' });
kitty.save().then(() => console.log('meow'));
```

Mongoose provides a straight-forward, **schema-based solution** to model your application data. It includes built-in type casting, validation, query building, business logic hooks and more, out of the box.



Get Professionally Supported Mongoose

## getting started

- [quick start guide](#)

## support

```

yann@pop-os:~/Documents/FORMATIONS/ANGULAR/sources/simplemean$ npm install --save mongoose

added 17 packages, and audited 998 packages in 10s

121 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
yann@pop-os:~/Documents/FORMATIONS/ANGULAR/sources/simplemean$

```

## Retour au code : Ajoutons un schéma pour le serveur

Dans backend/model/post.js

```

backend > models > JS post.js > [?] <unknown>
1  const mongoose = require('mongoose')
2  const postSchema = mongoose.Schema({
3    title: {type: String, required: true},
4    content: {type: String, required: true}
5  })
6
7  module.exports = mongoose.model('Post', postSchema)

```

Dans app.js

on ajoute le modèle et mongoose

<pre> SIMPLEMEAN &gt; .angular &gt; .vscode ✓ backend   ✓ models     JS post.js   JS app.js &gt; node_modules </pre>	<pre> backend &gt; JS app.js &gt; [?] mongoose 1  const express = require("express"); 2  const Post = require('./models/post') 3  const mongoose = require('mongoose') 4 5  const app = express(); 6  app.use(express.json()); 7 8 </pre>
--	---

Modifions app.post

```
app.post("/api/posts", (req, res, next) => {  
  console.log('====='+post)  
  const post = new Post({  
    title: req.body.title,  
    content: req.body.content  
  })  
  res.status(201).json({  
    message: 'Post added'  
  })  
  next()  
})
```

Testons

On voit un post dans console serveur

```
===== {  
  title: ';lk;lk',  
  content: ';lk;lk',  
  _id: new ObjectId('65bc96dec0e142971aa64b25')  
}
```

## Connectons nous à la base

Il faut donner les accès à express  
dans app.js

```
7 | mongoose.connect('mongodb+srv://creachyann:password@cluster0.sqgldfc.mongodb.net/?retryWrites=true&w=majority')  
8 | .then(() => console.log('Connected to atlas DB'))  
9 | .catch(() => console.log('Connection failed'))  
10 |
```

```
[nodemon] restarting due to changes...  
[nodemon] starting `node server.js`  
Connected to atlas DB
```



Bravo!

## Essayons de sauvegarder un post

Toujours dans app.js

```
22
23   app.post("/api/posts", (req, res, next) => {
24     //const post = req.body
25     const post = new Post({
26       title: req.body.title,
27       content: req.body.content
28     })
29     console.log(post)
30     post.save()
31     res.status(201).json({
32       message: 'Post added'
33     })
34     next()
35
36   })
37
```

Tester

# MyMessages

Post Title

Post Content

Save Post

title

title2

;lk;lk

lkjl

```
    {
      title: 'lkjl',
      content: 'kkljlkj',
      _id: new ObjectId('65bc98d853f8bb7a9cb56b46')
    }
  ]
}
```



Voir ses donnees

## Connect to Cluster0



### Connect to your application



#### Drivers

Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)



### Access your data through tools



#### Compass

Explore, modify, and visualize your data with MongoDB's GUI



#### Shell

Quickly add & update data using MongoDB's Javascript command-line interface



#### MongoDB for VS Code

Work with your data in MongoDB directly from your VS Code environment



#### Atlas SQL

Easily connect SQL tools to Atlas for data analysis and visualization



Go Back

Close

cliquer sur shell

## Connect to Cluster0



I don't have the MongoDB Shell installed

I have the MongoDB Shell installed

### 1. Select your operating system and download the MongoDB Shell

Ubuntu 20.04 ▼

Download mongosh (2.0.0)

or

Copy download URL

Mongosh(2.0.0) lets you connect to MongoDB to work with your data and configure your database. 2.0.0 or greater is required to work with Atlas Stream Processing

### 2. Add <your mongosh's download directory>/bin to your \$PATH variable. [How to](#)

### 3. Run your connection string in your command line

Use this connection string in your application

```
mongosh "mongodb+srv://cluster0.1fpy8xr.mongodb.net/" --apiVersion 1 --username
```

Installer au besoin

Copier la commande

Use this connection string in your application

```
mongosh "mongodb+srv://cluster0.1fpy8xr.mongodb.net/" --apiVersion 1 --username  
ycreach
```



```
yann@pop-os:~$ mongosh "mongodb+srv://cluster0.sqglfdc.mongodb.net/" --apiVersion 1 --username
creachyann
Enter password: *****
Current Mongosh Log ID: 65b155c298ecd5ea6a08767a
Connecting to:      mongodb+srv://<credentials>@cluster0.sqglfdc.mongodb.net/?appName=mong
osh+2.1.1
Using MongoDB:      6.0.13 (API Version 1)
Using Mongosh:      2.1.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-af9okz-shard-0 [primary] test> 
```

```
yann@pop-os:~$ mongosh "mongodb+srv://cluster0.sqglfdc.mongodb.net/" --apiVersion 1 --username
creachyann
Enter password: *****
Current Mongosh Log ID: 65b155c298ecd5ea6a08767a
Connecting to:      mongodb+srv://<credentials>@cluster0.sqglfdc.mongodb.net/?appName=mong
osh+2.1.1
Using MongoDB:      6.0.13 (API Version 1)
Using Mongosh:      2.1.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-af9okz-shard-0 [primary] test> use simplemeandb
switched to db simplemeandb
Atlas atlas-af9okz-shard-0 [primary] simplemeandb> 
```

```

Atlas atlas-af9okz-shard-0 [primary] simplemeandb> show collections
posts
Atlas atlas-af9okz-shard-0 [primary] simplemeandb> db.p
db.propertyIsEnumerable      db.printCollectionStats
db.printSecondaryReplicationInfo  db.printReplicationInfo
db.posts

Atlas atlas-af9okz-shard-0 [primary] simplemeandb> db.p
db.propertyIsEnumerable      db.printCollectionStats
db.printSecondaryReplicationInfo  db.printReplicationInfo
db.posts

Atlas atlas-af9okz-shard-0 [primary] simplemeandb> db.posts.find()
[
  {
    _id: ObjectId('65b154eb2dc0a2d5921071fe'),
    title: 'Test atlas',
    content: 'atlas con',
    __v: 0
  }
]
Atlas atlas-af9okz-shard-0 [primary] simplemeandb> 

```

Bravo!

## Le Get

Utilisons la méthode find

app.get devient:

```

44
45 // middleware
46 app.get("/api/posts", (req, res, next) => {
47   Post.find().then((documents) => {
48     res.status(200).json({
49       message: "Posts from server",
50       posts: documents,
51     });
52   });
53
54
55 });
56

```

Tester

The screenshot shows a web browser window with the URL `localhost:4200`. The page has a blue header with the text "MyMessages". Below the header is a form with two input fields: "Post Title" and "Post Content". Below these fields is a red button labeled "Save Post". Under the button, there are two dropdown menus, one showing "lkjl" and the other showing "atl".

To the right of the browser window is a terminal window titled "mongosh mongodb+srv://<credentials>@cluster0.sq". The terminal shows the following commands and output:

```

[
  {
    _id: ObjectId('65bc98d853f8bb7a9cb56b46'),
    title: 'lkjl',
    content: 'kkljlkj',
    __v: 0
  }
]
Atlas atlas-af9okz-shard-0 [primary] test> db.posts
[
  {
    _id: ObjectId('65bc98d853f8bb7a9cb56b46'),
    title: 'lkjl',
    content: 'kkljlkj',
    __v: 0
  },
  {
    _id: ObjectId('65bc9d8b07c85145b4828007'),
    title: 'atl',
    content: 'atl',
    __v: 0
  }
]
Atlas atlas-af9okz-shard-0 [primary] test>

```

Bravo !