

R5.A.10



Nouveaux paradigmes de bases de données



TP2 : accéder à MongoDB depuis Node.js

Objectif : comprendre comment connecter une application à MongoDB.

Equipes : projet à réaliser par groupes de 2 étudiants

Plan :

- Rappels MongoDB
- Présentation Node.js
- Connexion via mongodb / mongoose
- Requêtes courantes
- Framework Express (si besoin)
- Bonnes pratiques

MongoDB en quelques mots

Base orientée documents (JSON-like)

Stockage en collections

Requêtes riches en opérateurs

Idéal pour les applications web temps réel





Node.js et MongoDB

Node.js = JavaScript côté serveur

Asynchrone, orienté événements

Intégration naturelle avec MongoDB

Bibliothèques MongoDB pour Node.js

- mongodb (officiel) : bas niveau
- mongoose : ODM avec schémas
- Choix selon les besoins :
 - mongodb = contrôle
 - mongoose = confort



Installer les outils

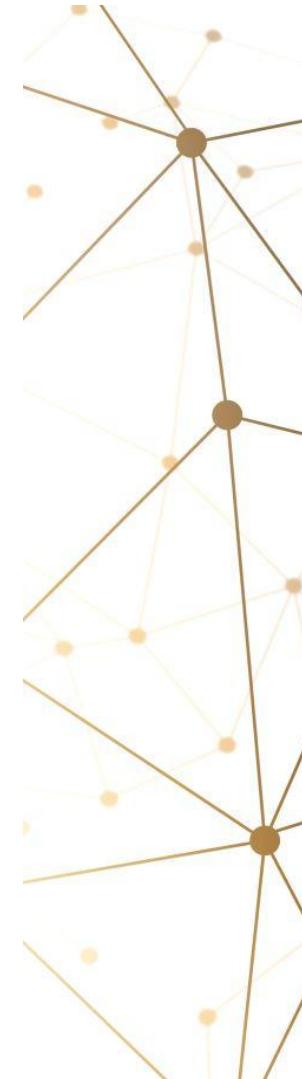
MongoDB local ou Atlas

Projet Node.js :

- `npm init -y`
- `npm install mongodb`

Ou avec Mongoose :

- `npm install mongoose`

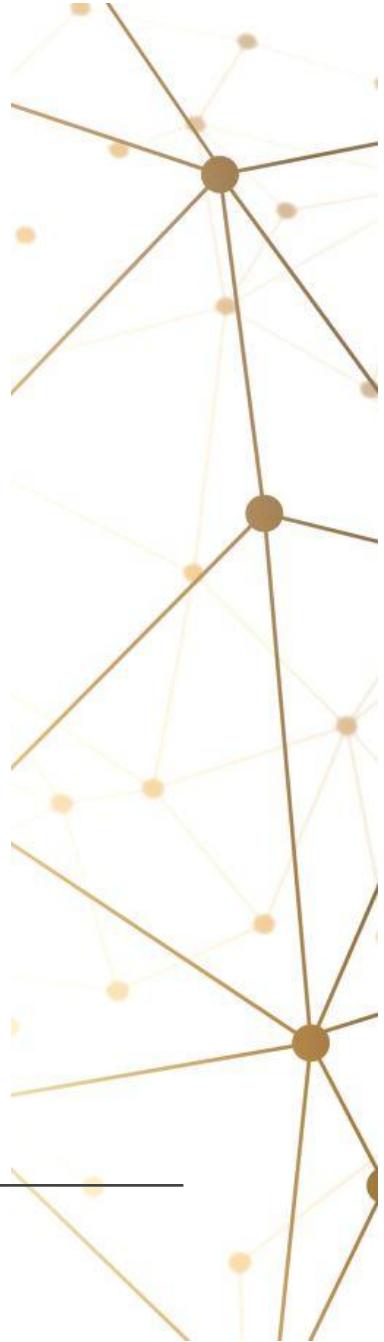


```
const { MongoClient } = require("mongodb");

const uri = "mongodb://localhost:27017";
const client = new MongoClient(uri);

async function run() {
  await client.connect();
  const db = client.db("maBase");
  const col = db.collection("maCollection");
  const docs = await col.find().toArray();
  console.log(docs);
  await client.close();
}
```

Connexion avec mongodb



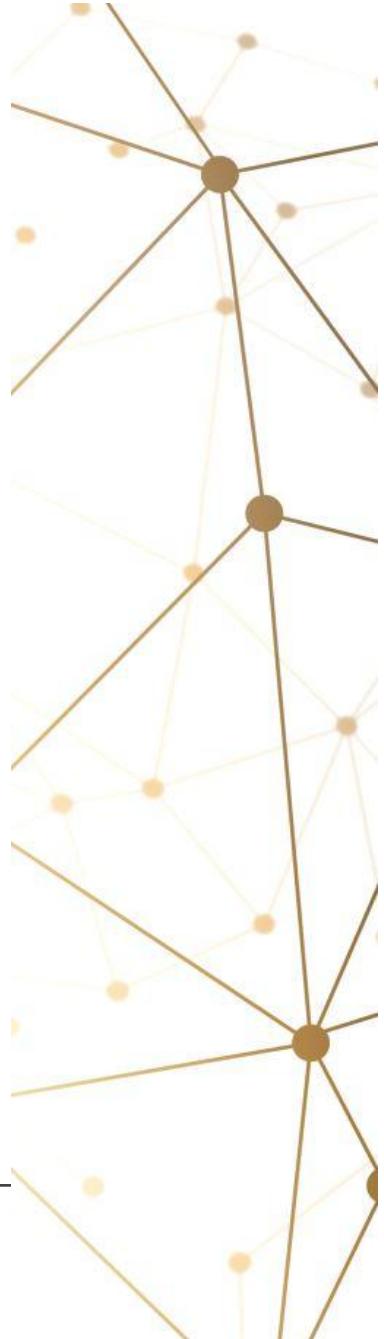
```
const mongoose = require("mongoose");

mongoose.connect("mongodb://localhost:27017/maBase");

const Film = mongoose.model("Film", {
  titre: String,
  annee: Number,
  genre: String
});

Film.find().then(console.log);
```

Connexion avec Mongoose



Requêtes courantes

- `insertOne`, `insertMany`
- `find`, `findOne`
- `updateOne`, `updateMany`
- `deleteOne`, `deleteMany`

Utilisation via `async/await`



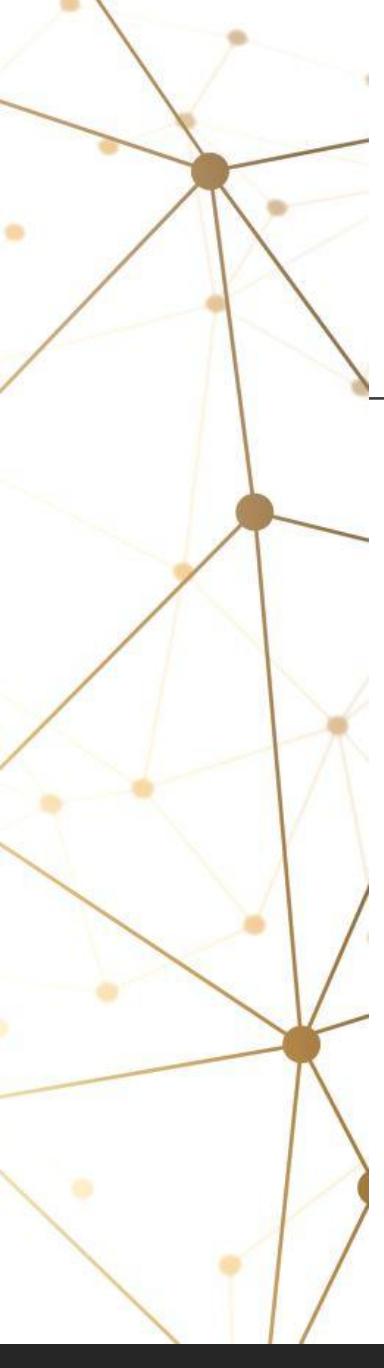
Express.js : un framework pour Node.js

- Framework minimaliste et flexible pour Node.js
- Permet de créer facilement des API REST
- Utilisé avec MongoDB pour créer des applications full-stack

Bonnes pratiques

- Utiliser async/await
- Utiliser try/catch pour les erreurs
- Penser à fermer la connexion
- Centraliser la configuration
- Utiliser les variables d'environnement (.env)





Ressources utiles

<https://www.mongodb.com/docs/drivers/node/current/>

<https://mongoosejs.com/>

Tutoriels :

- Connection : <https://welovedevs.com/fr/articles/nodejs-mongodb>
- Express : <https://welovedevs.com/fr/articles/express-js/>
- Créer une API : <https://medium.com/@saurabhraut3102/%EF%B8%8F-building-a-full-crud-application-with-node-js-and-mongodb-ca10d13d9537>
- Documentation officielle :
<https://www.mongodb.com/docs/drivers/node/current/>



À vous de jouer

Vous trouverez le cahier des charges ci-dessous :

[CdC Gestion de tâches.pdf](#)

Le résultat de votre travail devra être accessible dans un GIT
L'URL est à déposer dans le Moodle de cette ressource dans :

TP Appli gestion de tâches \ Livraisons