

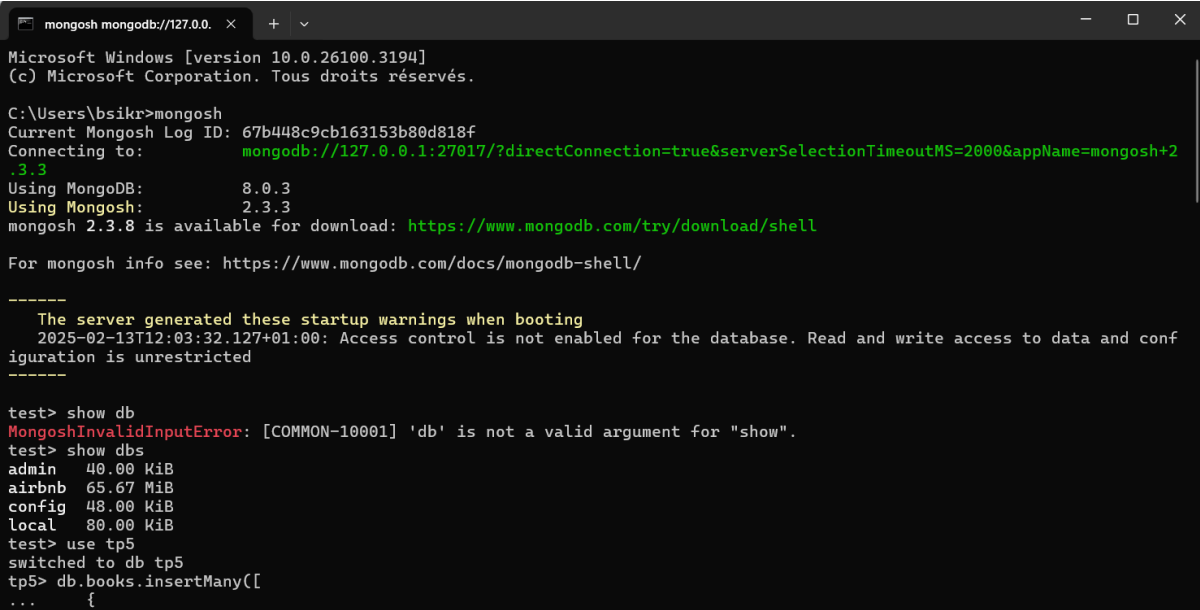
R6.A.05 Développement Avancé

TP5: Albums et Mongoose - 18/02/2025

Bsikri Mouhamed Nadir - 302

Etape 1:

mongodb était déjà installé sur ma machine.



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.3
Microsoft Windows [version 10.0.26100.3194]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\bsikr>mongosh
Current Mongosh Log ID: 67b448c9cb163153b80d818f
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.3
Using MongoDB:      8.0.3
Using Mongosh:       2.3.3
mongosh 2.3.8 is available for download: https://www.mongodb.com/try/download/shell

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

-----
  The server generated these startup warnings when booting
  2025-02-13T12:03:32.127+01:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
  -----

test> show db
MongoshInvalidInputError: [COMMON-10001] 'db' is not a valid argument for "show".
test> show dbs
admin    40.00 KiB
airbnb   65.67 MiB
config   48.00 KiB
local    80.00 KiB
test> use tp5
switched to db tp5
tp5> db.books.insertMany([
...    {
```

Etape 2 :

Nous avons installé fastify et mongoose grâce à la commande :

```
npm install mongoose fastify
```

Nous avons relié le service mongoDB à notre service NodeJS dans database/config.js :

```
import mongoose from "mongoose";

const MONGO_URI = "mongodb://localhost:27017/tp5";

export async function connectDB() {
  try {
    await mongoose.connect(MONGO_URI, {
      useNewUrlParser: true,
      useUnifiedTopology: true
    });
    console.log("✅ MongoDB connecté avec succès !");
  } catch (error) {
    console.error("❌ Erreur de connexion à MongoDB :", error);
    process.exit(1);
  }
}
```

Au sein de schemas/schema.js, nous avons défini le schéma de données en mettant "author" et "title" dans un tableau required.

Voici comment on a défini les schémas, avec "poche par défaut" :

```
format: {
  type: "string",
  enum: ["poche", "manga", "audio"],
  default: "poche"
}
```

Voici comment faire en sorte que ce qui est renvoyé n'est que l'auteur, le titre, la description ainsi que le format :

```
export const bookResponseSchema = {
  type: "object",
  properties: {
    title: { type: "string" },
    author: { type: "string" },
    description: { type: "string" },
    format: { type: "string" }
  }
};
```

Etape 3 :

Voici la requête qui récupère tous les livres :

The screenshot shows the Postman interface for a GET request to `https://localhost:3000/books/`. The request is successful, returning a 200 OK status with a response time of 8 ms and a body size of 886 B. The response body is a JSON array of three book objects.

Query Params

Key	Value	Description
Key	Value	Description

Body

```
13  },
14  {
15    "title": "Harry Potter à l'école des sorciers",
16    "author": "J.K. Rowling",
17    "description": "L'histoire d'un jeune sorcier découvrant son destin à Poudlard.",
18    "format": "poche"
19  },
20  {
21    "title": "1984",
22    "author": "George Orwell",
23    "description": "Un roman dystopique sur un monde sous surveillance constante.",
24    "format": "audio"
25  },
26  {
27    "title": "Nadir ???",
28    "author": "Mouhamed-Nadir",
29    "description": "Il était une fois, nadir",
30    "format": "poche"
31  }
32 ]
```

Voici comment récupérer un livre avec son id :

TP5 / <https://localhost:3000/books/id>

GET <https://localhost:3000/books/id> Send

Params • Authorization Headers (7) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Path Variables

Key	Value	Description	Bulk Edit
id	67b45533cb163153b80d8190	Description	

Body Cookies Headers (4) Test Results 200 OK • 39 ms • 322 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "title": "Le Seigneur des Anneaux",
3   "author": "J.R.R. Tolkien",
4   "description": "Une épopée fantastique dans un monde rempli de magie et d'anneaux de pouvoir.",
5   "format": "poche"
6 }
```

Voici comment ajouter un livre:

TP5 / <https://localhost:3000/books>

POST <https://localhost:3000/books> Send

Params Authorization Headers (10) Body • Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (4) Test Results 201 Created • 50 ms • 255 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "title": "Nadîr",
3   "author": "Mouhamed-Nadir",
4   "description": "Il était une fois, nadir",
5   "format": "poche"
6 }
```

Voici comment modifier le titre d'un livre:

TP5 / <https://localhost:3000/books/:id>

PUT <https://localhost:3000/books/:id>

Params • Authorization Headers (10) **Body** • Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "title" : "Nadir ???"
3 }
```

Body Cookies Headers (4) Test Results [↻](#) **200 OK** • 73 ms • 254 B

{} JSON ▾ ▷ Preview [Visualize](#) ▾

```
1 {
2   "title": "Nadir ???",
3   "author": "Mouhamed-Nadir",
4   "description": "Il était une fois, nadir",
5   "format": "poche"
6 }
```

Voici comment supprimer un livre :

TP5 / <https://localhost:3000/books/:id> [Save](#) [Share](#)

DELETE <https://localhost:3000/books/:id> **Send** ▾

Params • Authorization Headers (8) **Body** • Scripts Settings [Cookies](#)

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Path Variables

	Key	Value	Description	...	Bulk Edit
	id	67b45533cb163153b80d8194	Description		

Body Cookies Headers (4) Test Results [↻](#) **200 OK** • 36 ms • 189 B [Save Response](#) [↻](#) [📄](#) [🔍](#) [🔗](#)

{} JSON ▾ ▷ Preview [Visualize](#) ▾

```
1 {
2   "message": "Livre supprimé avec succès"
3 }
```