Prenom : **Kamal** Nom : **ELMAGOURI**
Filière : **MUP IWMBigData M2**

# MINI PROJECT NoSQL (MongoDB)

## 1. Intorduction

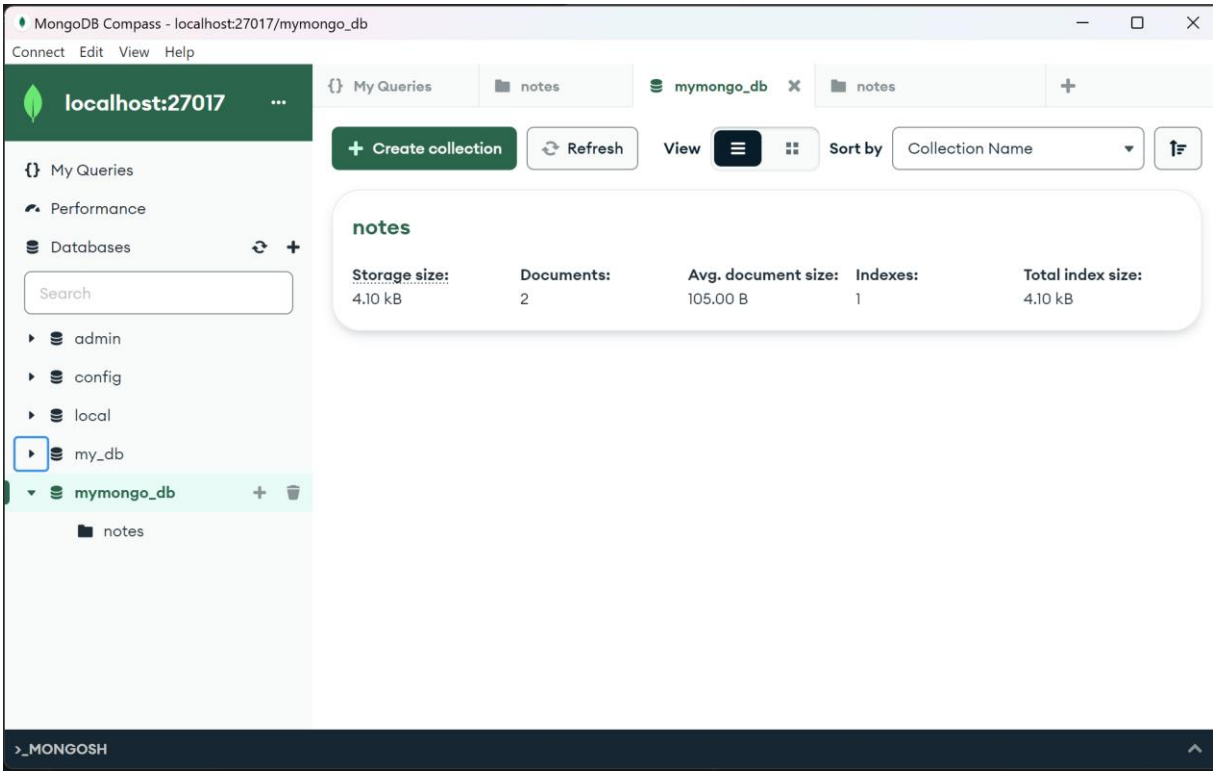
L'objectif principal du projet était de créer une application utilisant **Node.js** et **Express.js** pour fournir des **API CRUD** (**C**reate, **R**ead, **U**pdate, **D**elete) pour gérer des notes avec un titre et un corps, en utilisant une base de données **MongoDB** pour le stockage.

## 2. Configuration de l'environnement de développement :

- **Node.js** et **MongoDB** ont été installés sur le système de développement.
- Le projet **Node.js** a été initialisé en utilisant **npm**.
- **Express.js** a été configuré en tant que framework web pour l'application.

## 3. Mise en place de la base de données MongoDB :

- Une base de données MongoDB a été créée.
- Une collection "notes" a été mise en place pour stocker les notes.
- Mongoose a été utilisé pour établir une connexion entre l'application Node.js et MongoDB.




**NB : Une Fois j'ai Définir le nom de la base de donnée dans le fichier de configuration du projet.La collection notes à été crée automatiquement**

## 4. Création des API CRUD :

- ✓ **Create (POST) :**
  - Une route POST a été mise en place pour permettre la création de nouvelles notes.

```
//La route
noteRoutes.post('/notes/add', addNote); // La route

//le controller
 const addNote = (req, res) => {
     const {title, body} = req.body

     const newNote = new Note({ title, body });

     newNote.save().then(() => {
         res.json("note added");
     }).catch(err => console.log(err));
 }
```

  - Les données entrantes ont été validées avant d'être enregistrées dans la base de données.
- ✓ **Read (GET) :**
  - Une route GET a été implémentée pour récupérer toutes les notes ou une note spécifique.

```
//la route
noteRoutes.get('/notes', getAllNotes);
noteRoutes.get('/notes/:id', noteById);

//Le controller
const getAllNotes = async (req, res) => {
    try {
        const notes = await Note.find().select('title body createdAt');
        res.json({
            notesData : notes
        });
    } catch (error) {
        res.status(500).json({ message: err.message });
    }
}
const noteById = (req, res) => {
    Note.findById(req.params.id).then(note => {
        res.json({ note });
    }).catch(err => console.log(err));
}
```

✓ **Update (PUT) :**

- Une route PUT a été créée pour mettre à jour une note existante.
- Les données entrantes ont été validées avant la mise à jour de la note dans la base de données.

```
La Route
noteRoutes.put('/notes/:id', updateNote);

const updateNote = (req, res) => {
    Note.findById(req.params.id).then(note => {
        const {title, body} = req.body;
        note.title = title;
        note.body = body;
        note.save().then(() => {
            res.json("note updated");
        }).catch(err => console.log(err));
    }).catch(err => console.log(err));
}
```

✓ **Delete (DELETE) :**

- Une route DELETE a été établie pour permettre la suppression d'une note.

```
noteRoutes.delete('/notes/:id', deleteNote);

const deleteNote = (req, res) => {
   Note.findByIdAndDelete(req.params.id).then(note => {
      res.json("note deleted");
   }).catch(err => console.log(err));
}
```

- La note a été supprimée de la base de données.

## 5. Tests :

Des tests unitaires et/ou d'intégration ont été développés pour garantir le bon fonctionnement des API CRUD.

L'outil **Postman** a été utilisé pour tester manuellement les différentes fonctionnalités de l'API.

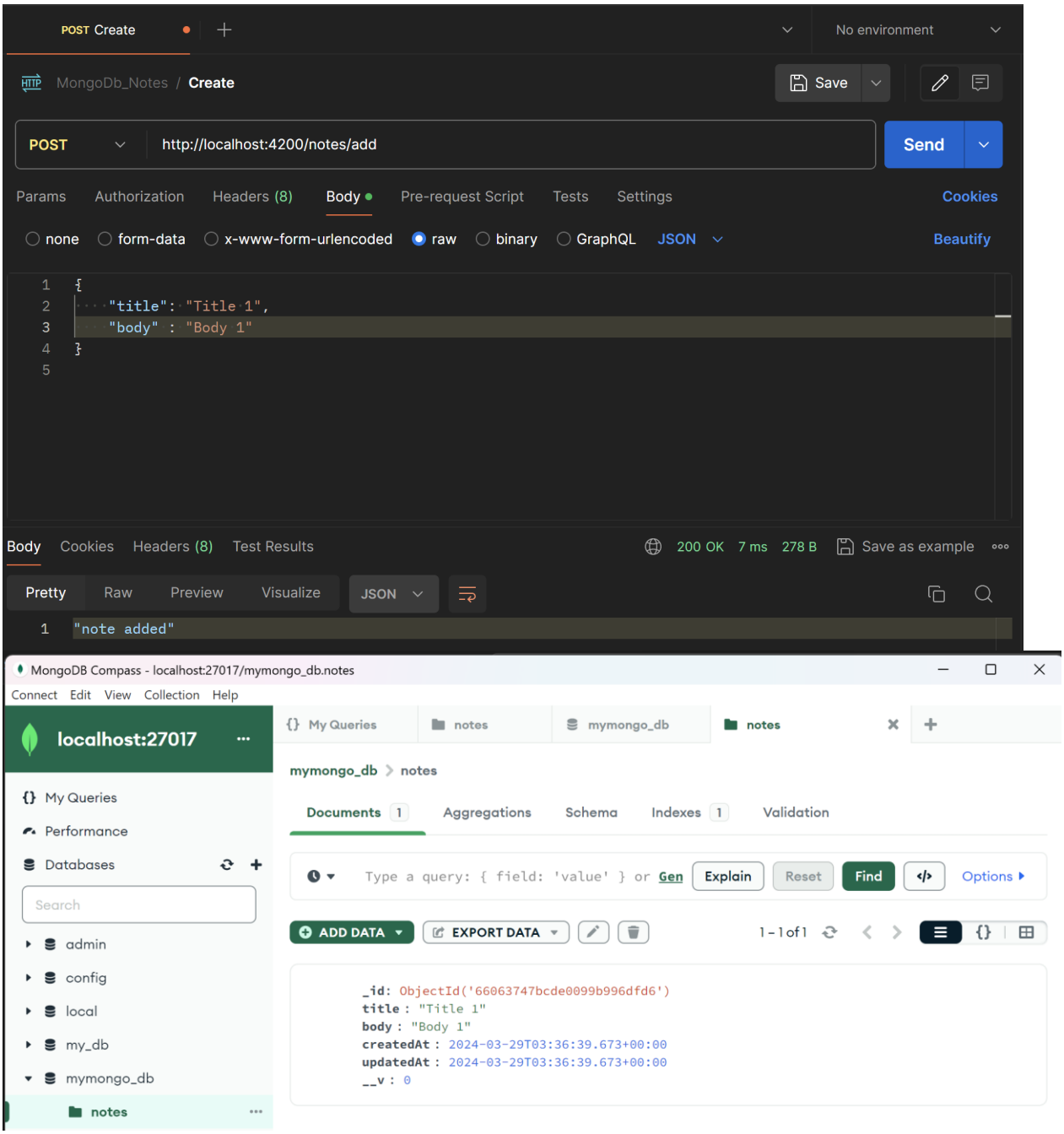
## 6. Conclusion :

- Le projet a été réalisé avec succès, fournissant des **API CRUD** fonctionnelles pour gérer des notes dans une base de données **MongoDB**.

# 7. Annexes :

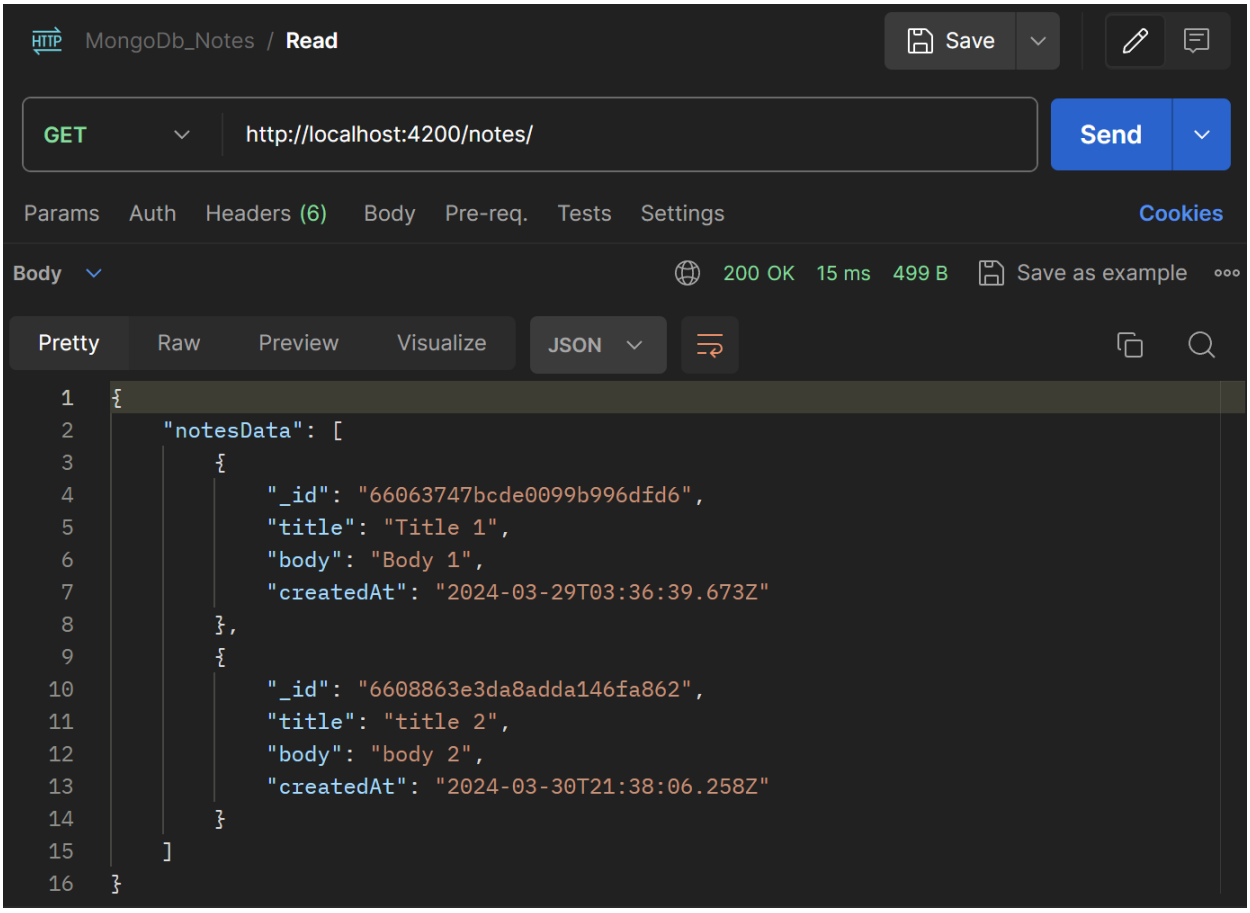
- Le code source de l'application, y compris le fichier de configuration, les routes et le contrôleur
- Des captures d'écran de l'application en cours d'exécution et des tests effectués avec **Postman**. et vérifier le contenu avec le **mongoDb Compass**

## Create

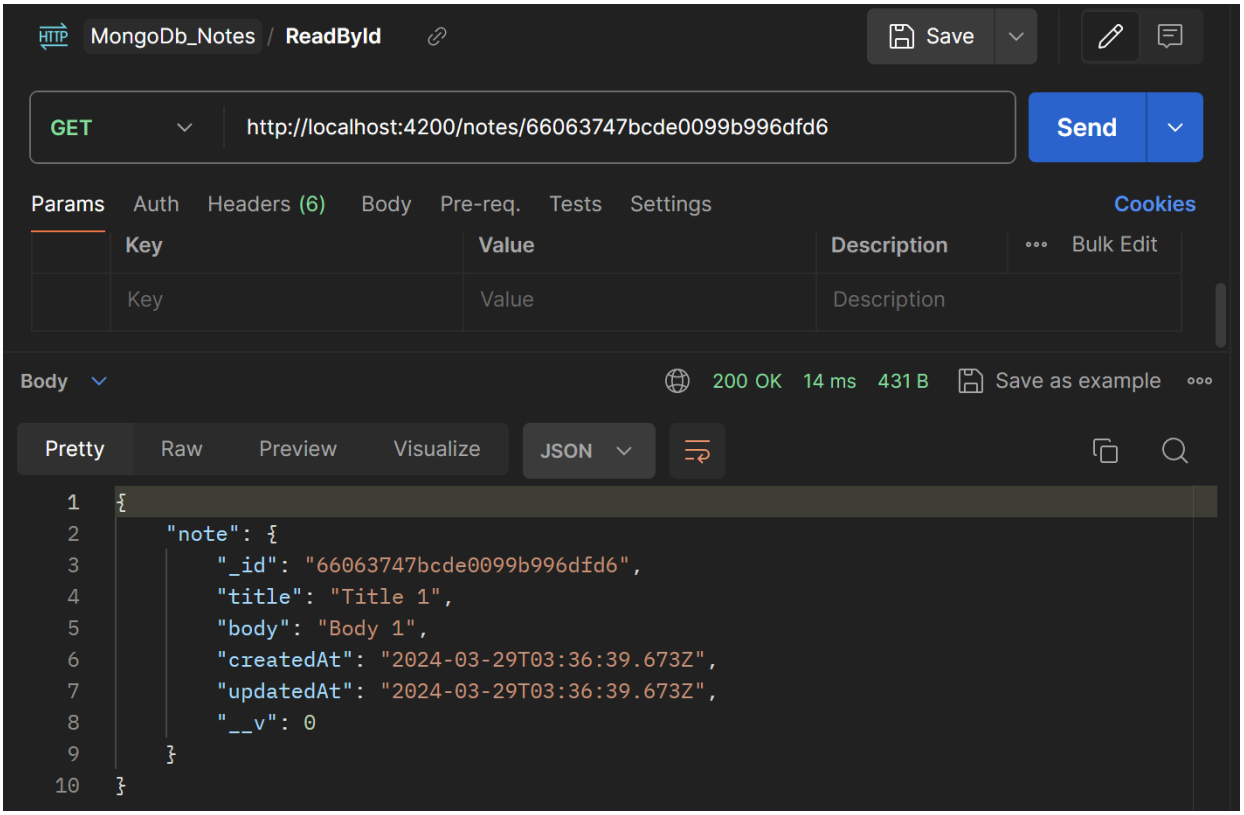


## Read

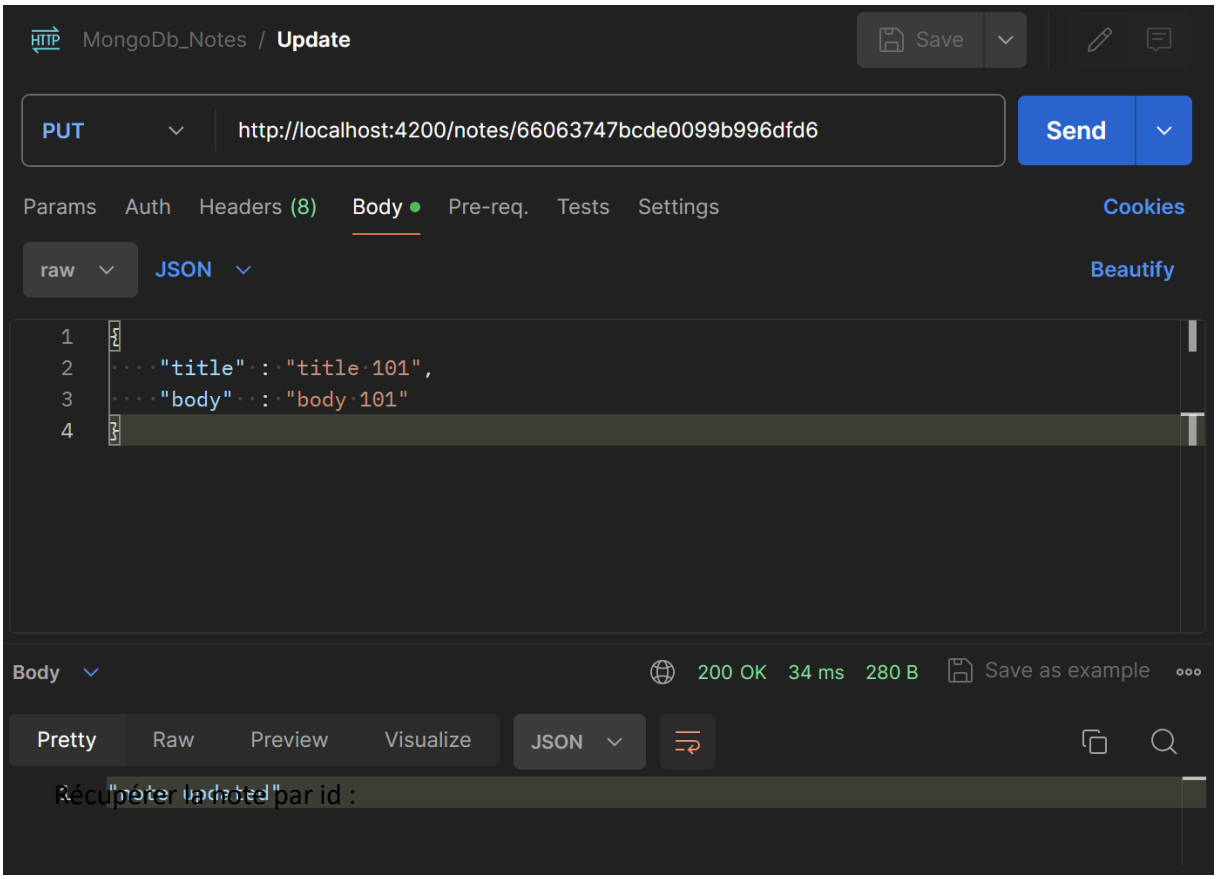
- Récupérer tous les notes




- Récupérer une note par l'identifiant ID

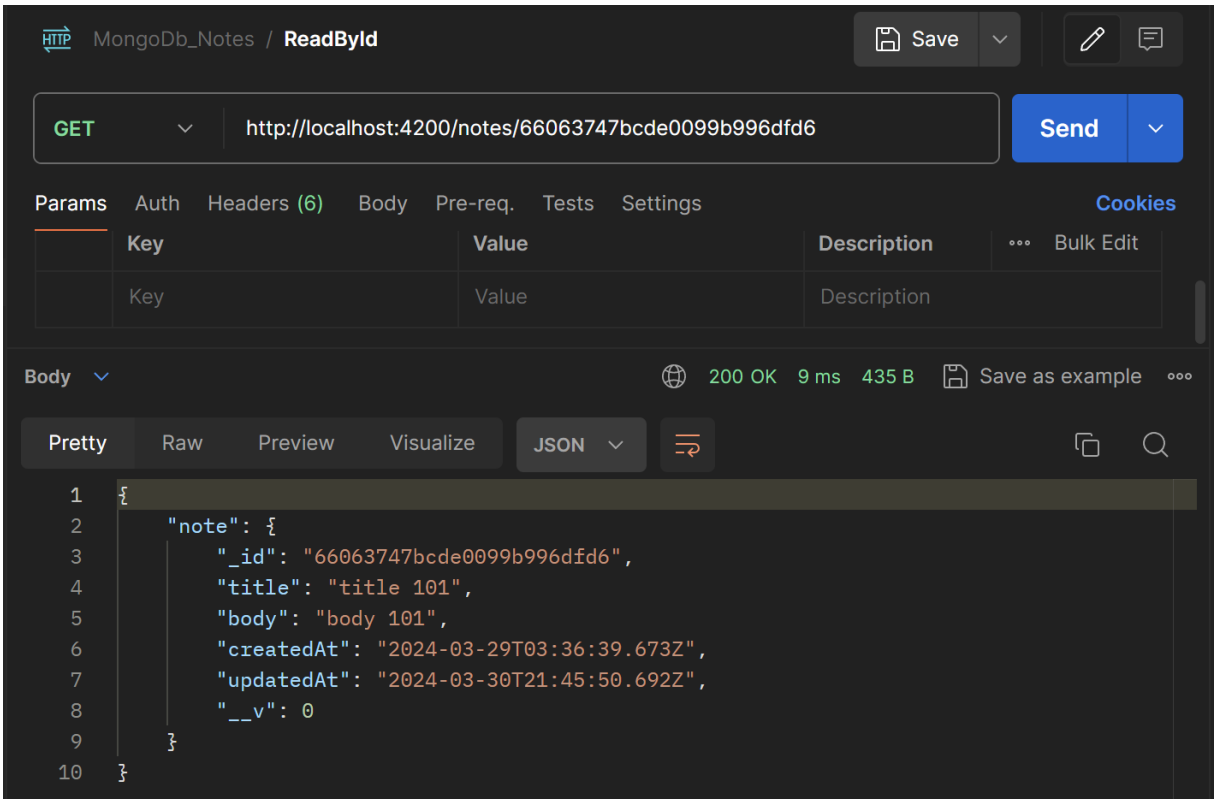


## Update

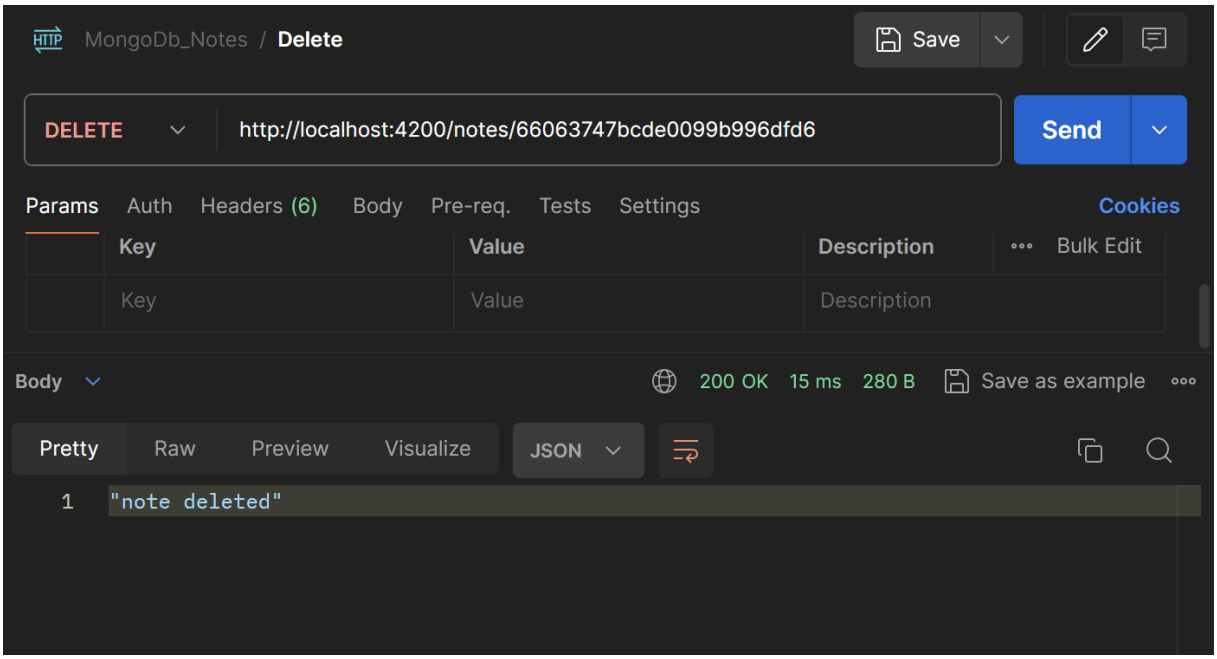



Récupérer la note par id :

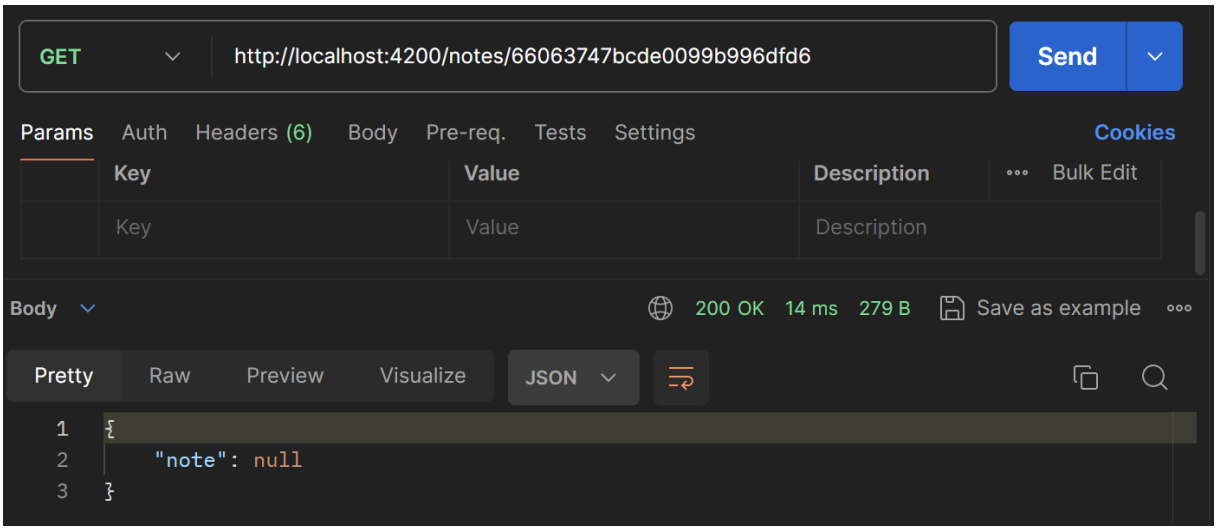
- Récupérer une note par l'identifiant ID



## Delete




- Récupérer une note par l'identifiant ID




- Récupérer une note par l'identifiant ID