

Chapitre 3

Gestion des données

Partie 2

CRUD en No SQL

NoSQL

- NoSQL ou “Not Only SQL”, ce sont des bases de données :
 - a. Sans schéma strict (plus flexibles)
 - b. Adaptées aux gros volumes
 - c. Stockent souvent les données en documents JSON, par exemple dans MongoDB

```
{
  "books": [
    {
      "id": "01",
      "language": "Java",
      "author": "H. Javeson",
      "year": 2015
    },
    {
      "id": "07",
      "language": "C++",
      "edition": "second",
      "author": "E. Sepp",
      "price": 22.25
    }
  ]
}
```

MongoDB

- Base de données **NoSQL orientée documents**
- Stockage des données en JSON
- Idéal pour :
 - a. Données non structurées ou souvent modifiées
 - b. Applications en temps réel



Mongoose

- ODM (Object Data Modeling), équivalent ORM pour NoSQL
- Interface entre Node et MongoDB
- Permet de définir un schéma pour structurer les données
- Fournit une gestion automatisée de :
 - Validation
 - Middleware
 - Relations
- Simplifie les requêtes avec des fonctions prêtes à l'emploi (find(), create())... voir documentation)

Connexion avec Mongoose et MongoDB Atlas



Uploaded using RayThis Extension

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

mongoose.connect('mongodb://localhost:27017/test').then(() => {
  console.log('Connected to MongoDB');
}).catch(err => {
  console.error('Error connecting to MongoDB:', err);
});
```

Définir un schéma

```
Uploaded using RayThis Extension

const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
  name: String,
  price: Number,
  quantity: Number,
});

module.exports = mongoose.model('Product', productSchema);
```

Faire des requêtes

```
Uploaded using RayThis Extension

const getAll = (req, res) => {
  Product.find().then(products => {
    res.status(200).json(products);
  }).catch(err => {
    res.status(500).json(err);
  });
};

const getById = (req, res) => {
  const { id } = req.params;
  Product.findById(id).then(product => {
    if (!product) {
      res.status(404).json({ message: 'No product found' });
    } else {
      res.status(200).json(product);
    }
  }).catch(err => {
    res.status(500).json(err);
  });
};

const create = (req, res) => {
  const { name, price, quantity } = req.body;
  const product = new Product({ name, price, quantity });
  product.save().then(product => {
    res.status(201).json(product);
  }).catch(err => {
    res.status(500).json(err);
  });
};
```


Faire des requêtes

```
Uploaded using RayThis Extension

const update = (req, res) => {
  const { id } = req.params;
  const { name, price, quantity } = req.body;
  Product.findByIdAndUpdate(id, { name, price, quantity }, { new: true }).then(product => {
    if (!product) {
      res.status(404).json({ message: 'No product found' });
    } else {
      res.status(200).json(product);
    }
  }).catch(err => {
    res.status(500).json(err);
  });
};

const deleteById = (req, res) => {
  const { id } = req.params;
  Product.findByIdAndDelete(id).then(product => {
    if (!product) {
      res.status(404).json({ message: 'No product found' });
    } else {
      res.status(200).json(product);
    }
  }).catch(err => {
    res.status(500).json(err);
  });
};
```

Exercices



TP 2 : Validation des acquis