**🟢 Day 4: Add MongoDB to Your Express Backend**

**adding MongoDB to your backend** so your todos are saved even if the server restarts

**1. Set Up MongoDB**

**a) Install MongoDB Locally (Optional)**

bash

mongod

**b) Or Use MongoDB Atlas (Recommended & Free)**

* Go to [MongoDB Atlas](https://www.mongodb.com/cloud/atlas/register), create a free cluster.
* Create a database user and get your connection string

**2. Install Mongoose in Your Backend**

From your mern-backend folder:

npm install mongoose

**3. Connect Express to MongoDB**

At the top of your index.js:

const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost:27017/mern-todos', {

useNewUrlParser: true,

useUnifiedTopology: true,

})

.then(() => console.log('MongoDB connected!'))

.catch(err => console.error('MongoDB connection error:', err));

*If using Atlas, replace the connection string with your Atlas URI.*

**4. Create a Todo Model**

Add this after your mongoose.connect:

const todoSchema = new mongoose.Schema({

text: String,

done: Boolean,

});

const Todo = mongoose.model('Todo', todoSchema);

**5. Update Your API Endpoints to Use MongoDB**

Replace your in-memory array with MongoDB calls:

// Get all todos

app.get('/todos', async (req, res) => {

const todos = await Todo.find();

res.json(todos);

});

// Add a todo

app.post('/todos', async (req, res) => {

const todo = new Todo(req.body);

await todo.save();

res.status(201).json(todo);

});

// Delete a todo by id

app.delete('/todos/:id', async (req, res) => {

await Todo.findByIdAndDelete(req.params.id);

res.status(204).end();

});

// Toggle done

app.put('/todos/:id', async (req, res) => {

const todo = await Todo.findById(req.params.id);

todo.done = !todo.done;

await todo.save();

res.json(todo);

});

**6. Update Your Frontend to Use MongoDB IDs**

* Todos now have \_id instead of array index.
* Update React code to use \_id for delete/toggle:

const deleteTodo = (id) => {

fetch(`http://localhost:5000/todos/${id}`, { method: 'DELETE' })

.then(() => setTodos(todos.filter(todo => todo.\_id !== id)));

};

// Example for toggle

const toggleDone = (id) => { //use id instead of index

fetch(`http://localhost:5000/todos/${id}`, { method: 'PUT' }) //use id

.then(res => res.json())

.then(updatedTodo => { //use map here unlike step3

setTodos(todos.map(todo => todo.\_id === id ? updatedTodo : todo)); //

});

};

* When rendering todos:

{todos.map(todo => (

<li key={todo.\_id}> //use in click handlers

{todo.done ? 'completed=> ' : 'yet to do => '}{todo.text}

<button onClick={() => toggleDone(todo.\_id)}>toggle</button>

<button onClick={() => deleteTodo(todo.\_id)}>Delete</button> //use todo.\_id

</li>

))}

**7. Test Everything**

* Add, delete, and toggle todos—refresh the page and see your data persist!
* Use [MongoDB Compass](https://www.mongodb.com/products/compass) or Atlas dashboard to view your data.