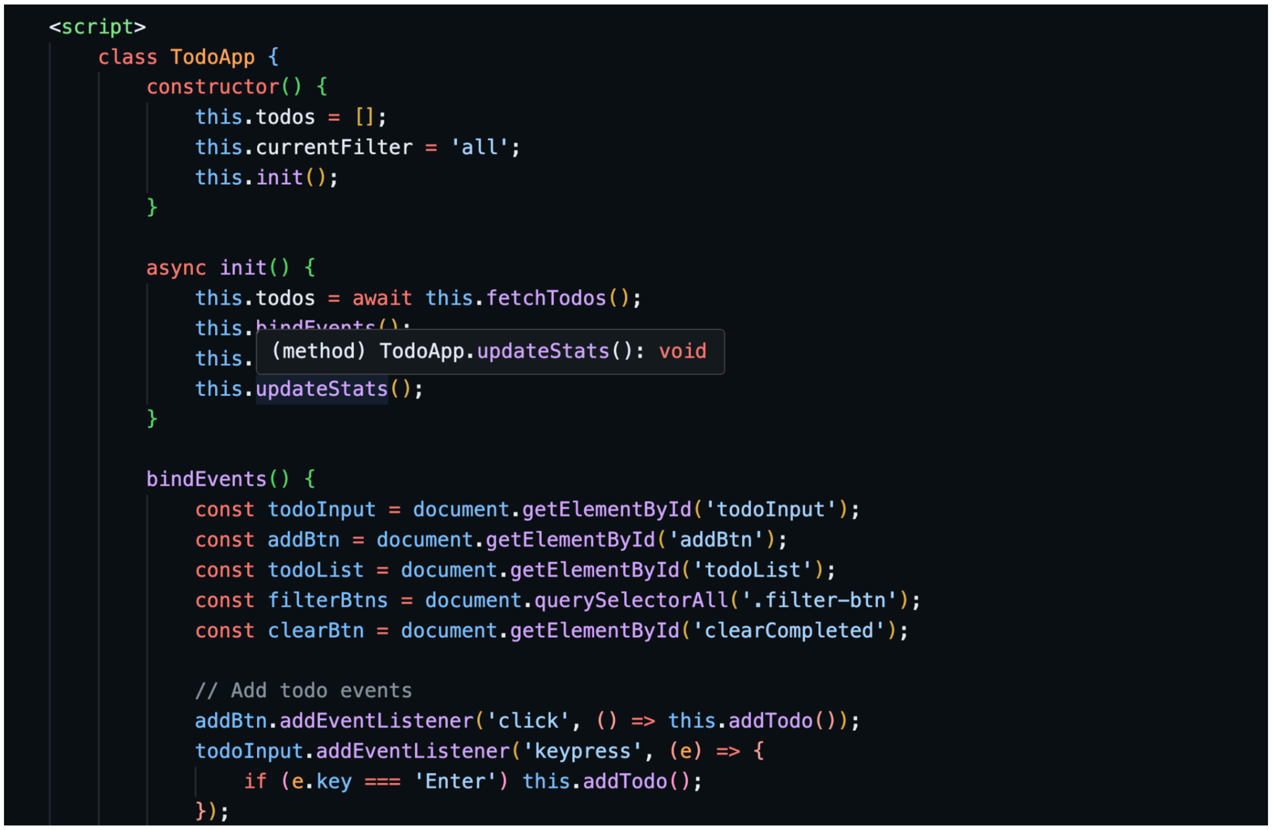
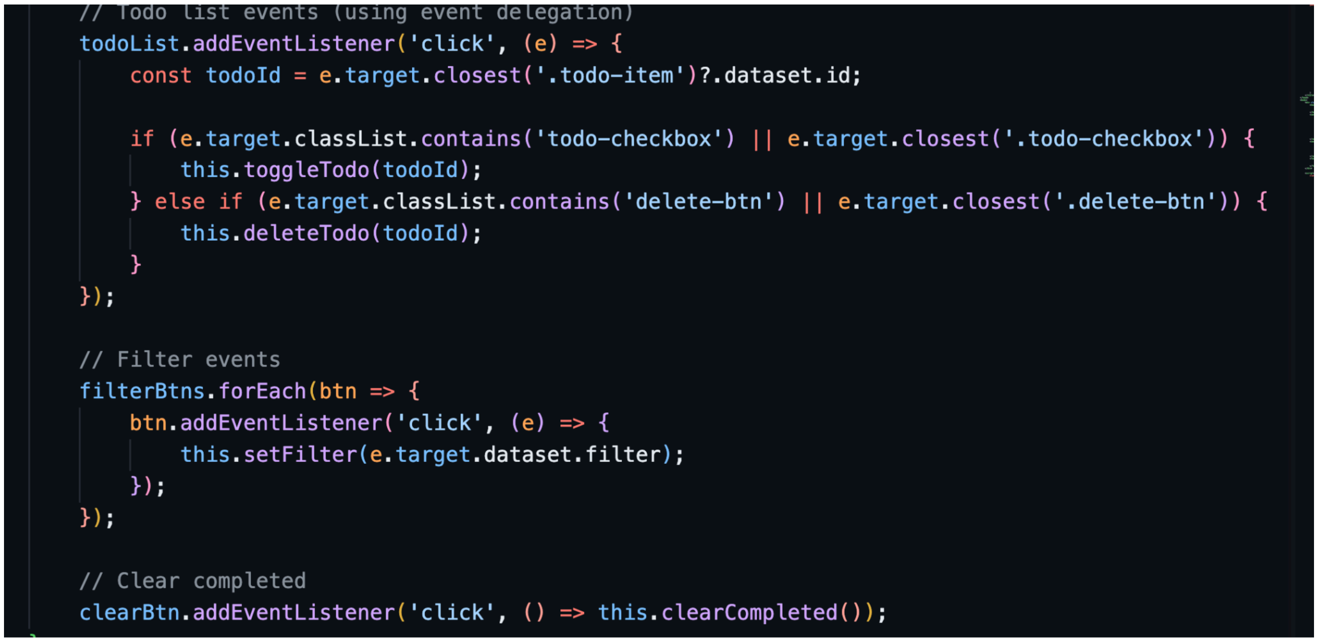
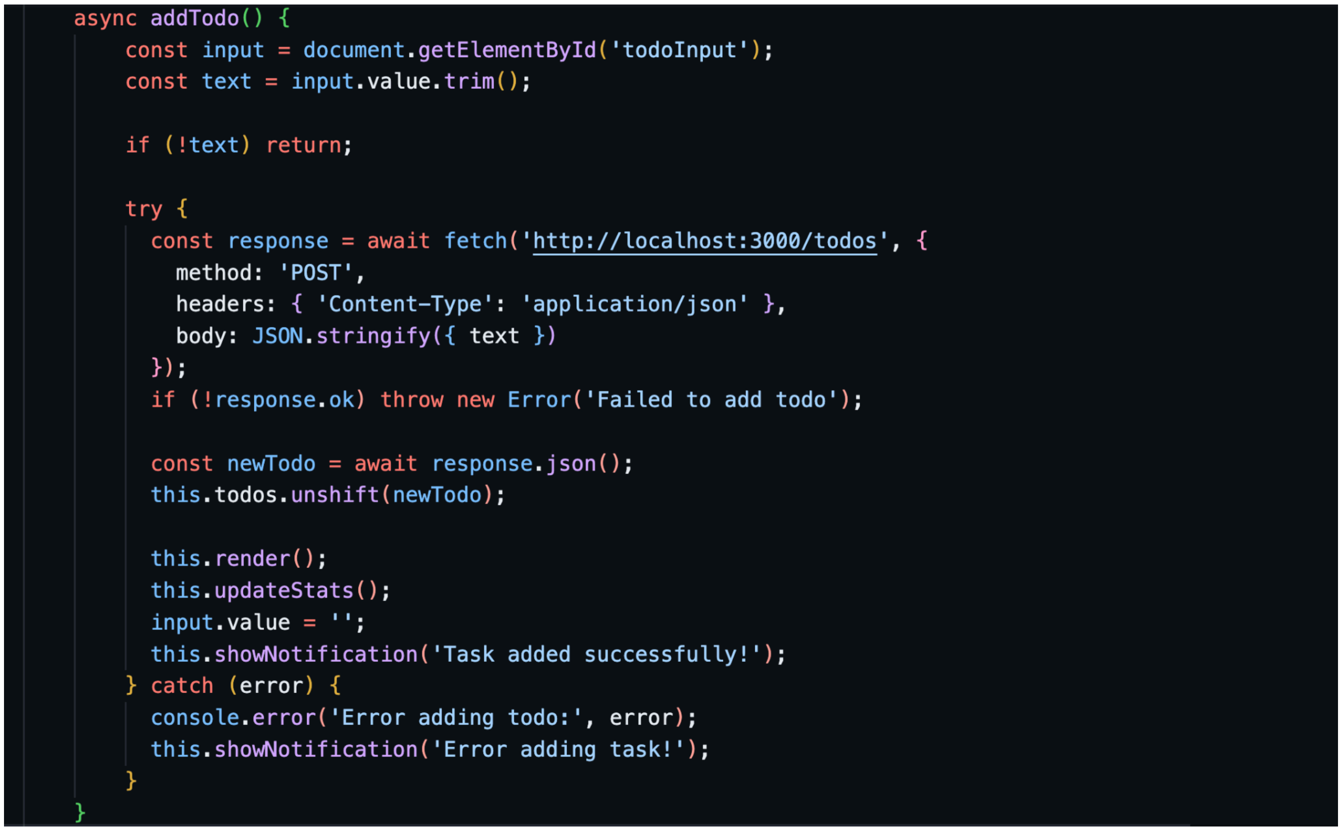
Source code :



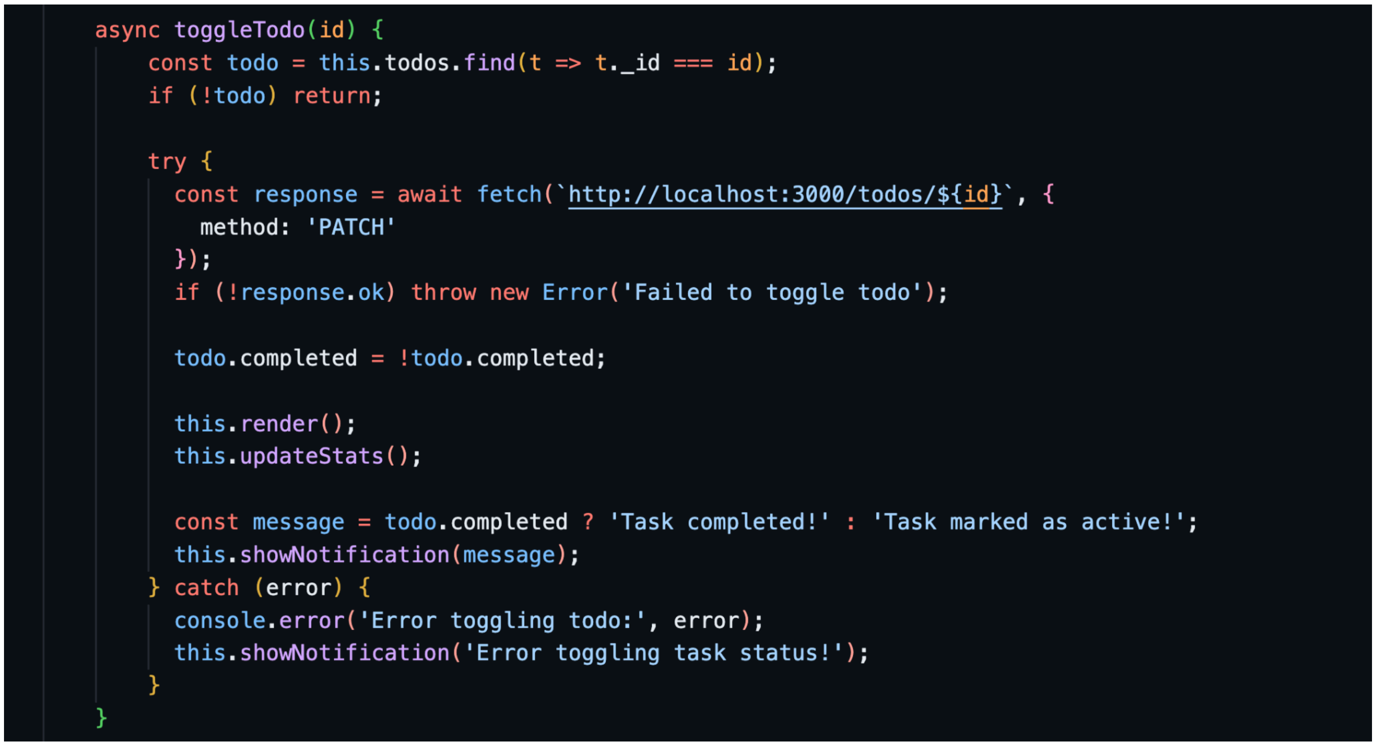
```
<script>
    class TodoApp {
        constructor() {
            this.todos = [];
            this.currentFilter = 'all';
            this.init();
        }

        async init() {
            this.todos = await this.fetchTodos();
            this.bindEvents();
            this. (method) TodoApp.updateStats(): void
            this.updateStats();
        }

        bindEvents() {
            const todoInput = document.getElementById('todoInput');
            const addBtn = document.getElementById('addBtn');
            const todoList = document.getElementById('todoList');
            const filterBtns = document.querySelectorAll('.filter-btn');
            const clearBtn = document.getElementById('clearCompleted');

            // Add todo events
            addBtn.addEventListener('click', () => this.addTodo());
            todoInput.addEventListener('keypress', (e) => {
                if (e.key === 'Enter') this.addTodo();
            });
```



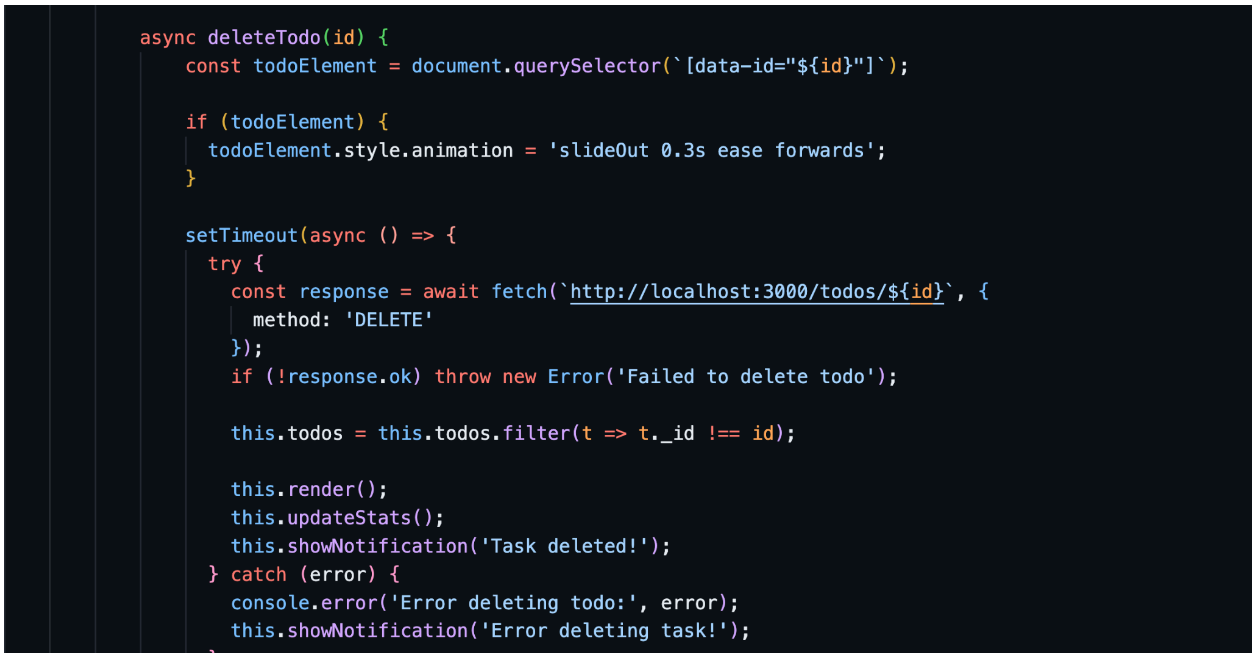
```
    // Todo list events (using event delegation)
    todoList.addEventListener('click', (e) => {
        const todoId = e.target.closest('.todo-item')?.dataset.id;

        if (e.target.classList.contains('todo-checkbox') || e.target.closest('.todo-checkbox')) {
            this.toggleTodo(todoId);
        } else if (e.target.classList.contains('delete-btn') || e.target.closest('.delete-btn')) {
            this.deleteTodo(todoId);
        }
    });

    // Filter events
    filterBtns.forEach(btn => {
        btn.addEventListener('click', (e) => {
            this.setFilter(e.target.dataset.filter);
        });
    });

    // Clear completed
    clearBtn.addEventListener('click', () => this.clearCompleted());
```



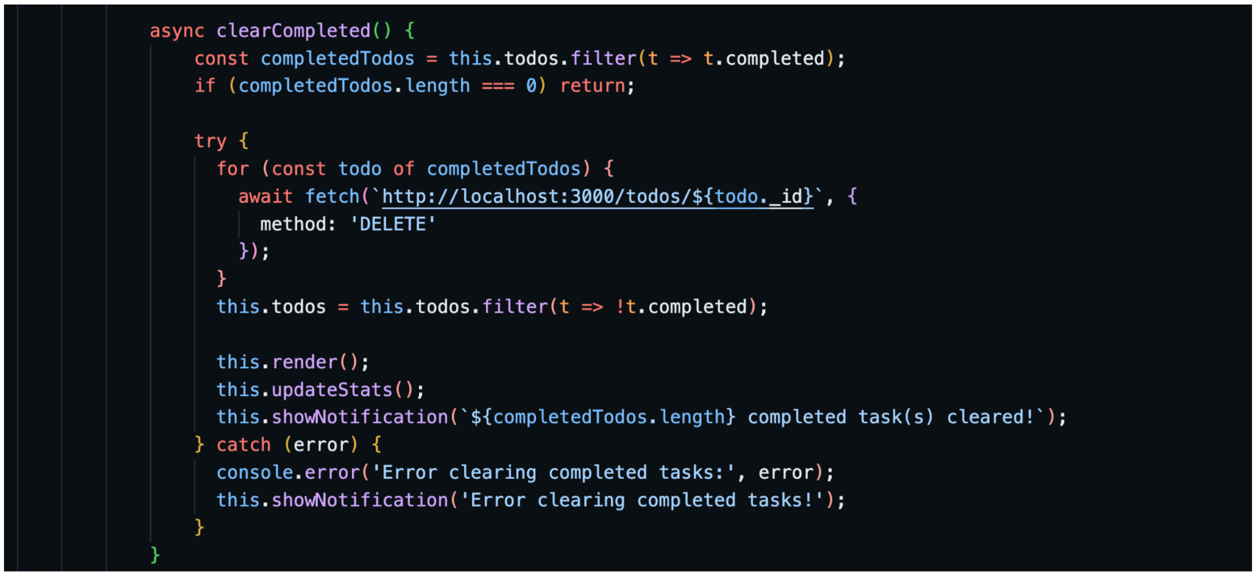
```javascript
async addTodo() {
    const input = document.getElementById('todoInput');
    const text = input.value.trim();

    if (!text) return;

    try {
      const response = await fetch('http://localhost:3000/todos', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ text })
      });
      if (!response.ok) throw new Error('Failed to add todo');

      const newTodo = await response.json();
      this.todos.unshift(newTodo);

      this.render();
      this.updateStats();
      input.value = '';
      this.showNotification('Task added successfully!');
    } catch (error) {
      console.error('Error adding todo:', error);
      this.showNotification('Error adding task!');
    }
}
```



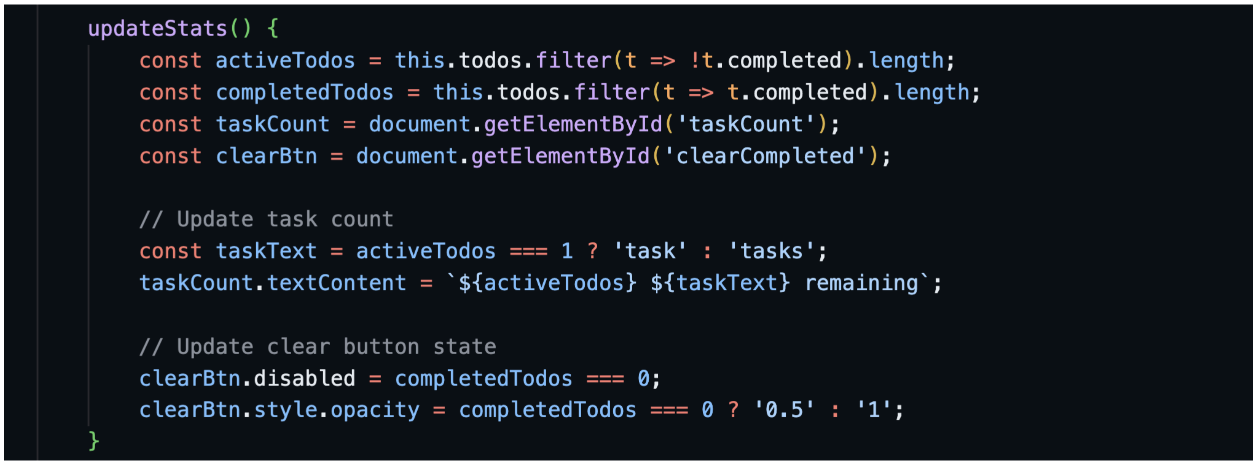
```javascript
async toggleTodo(id) {
    const todo = this.todos.find(t => t._id === id);
    if (!todo) return;

    try {
      const response = await fetch(`http://localhost:3000/todos/${id}`, {
        method: 'PATCH'
      });
      if (!response.ok) throw new Error('Failed to toggle todo');

      todo.completed = !todo.completed;

      this.render();
      this.updateStats();

      const message = todo.completed ? 'Task completed!' : 'Task marked as active!';
      this.showNotification(message);
    } catch (error) {
      console.error('Error toggling todo:', error);
      this.showNotification('Error toggling task status!');
    }
}
```



```
async deleteTodo(id) {
    const todoElement = document.querySelector(`[data-id="${id}"]`);

    if (todoElement) {
      todoElement.style.animation = 'slideOut 0.3s ease forwards';
    }

    setTimeout(async () => {
      try {
        const response = await fetch(`http://localhost:3000/todos/${id}`, {
          method: 'DELETE'
        });
        if (!response.ok) throw new Error('Failed to delete todo');

        this.todos = this.todos.filter(t => t._id !== id);

        this.render();
        this.updateStats();
        this.showNotification('Task deleted!');
      } catch (error) {
        console.error('Error deleting todo:', error);
        this.showNotification('Error deleting task!');
```

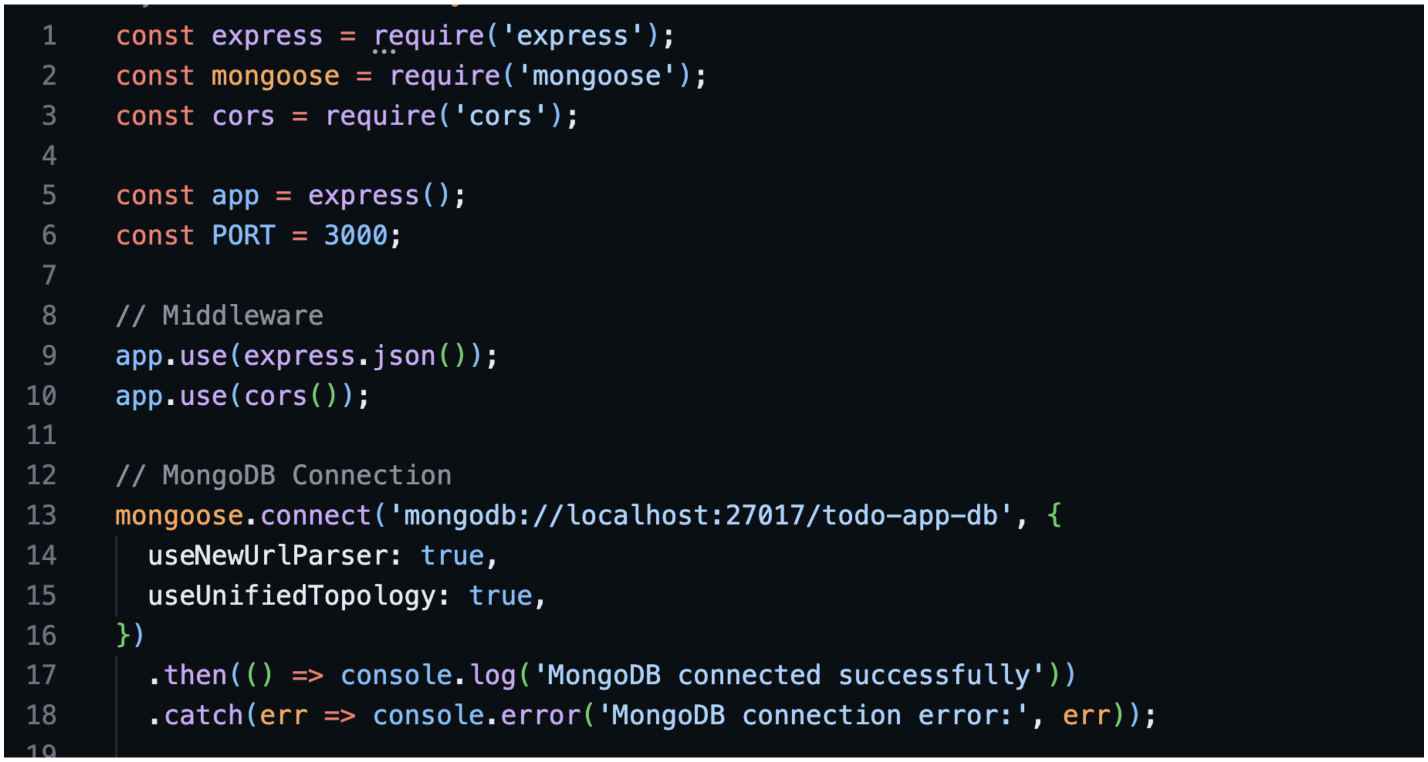


```
async clearCompleted() {
    const completedTodos = this.todos.filter(t => t.completed);
    if (completedTodos.length === 0) return;

    try {
      for (const todo of completedTodos) {
        await fetch(`http://localhost:3000/todos/${todo._id}`, {
          method: 'DELETE'
        });
      }
      this.todos = this.todos.filter(t => !t.completed);

      this.render();
      this.updateStats();
      this.showNotification(`${completedTodos.length} completed task(s) cleared!`);
    } catch (error) {
      console.error('Error clearing completed tasks:', error);
      this.showNotification('Error clearing completed tasks!');
    }
}
```



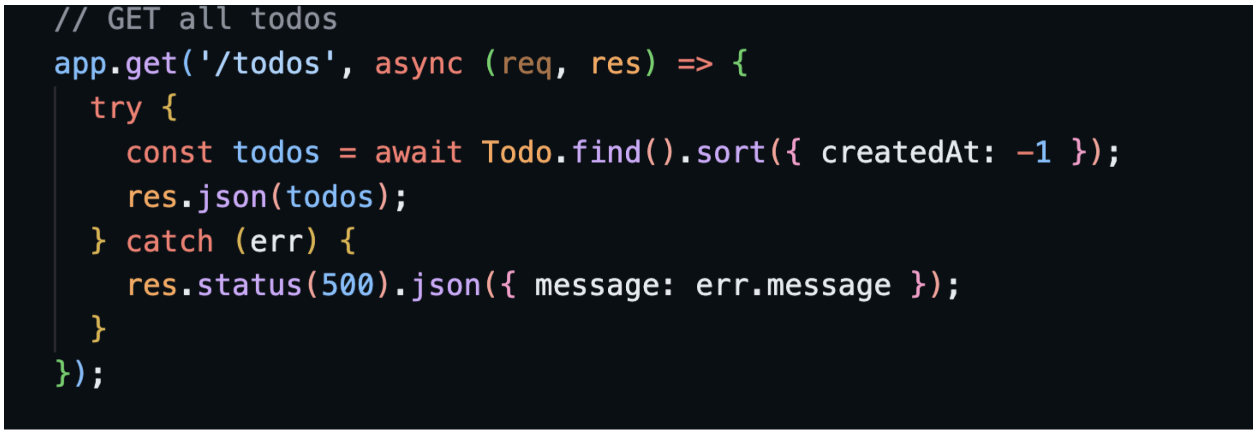
```
updateStats() {
    const activeTodos = this.todos.filter(t => !t.completed).length;
    const completedTodos = this.todos.filter(t => t.completed).length;
    const taskCount = document.getElementById('taskCount');
    const clearBtn = document.getElementById('clearCompleted');

    // Update task count
    const taskText = activeTodos === 1 ? 'task' : 'tasks';
    taskCount.textContent = `${activeTodos} ${taskText} remaining`;

    // Update clear button state
    clearBtn.disabled = completedTodos === 0;
    clearBtn.style.opacity = completedTodos === 0 ? '0.5' : '1';
}
```

MongoDB connection

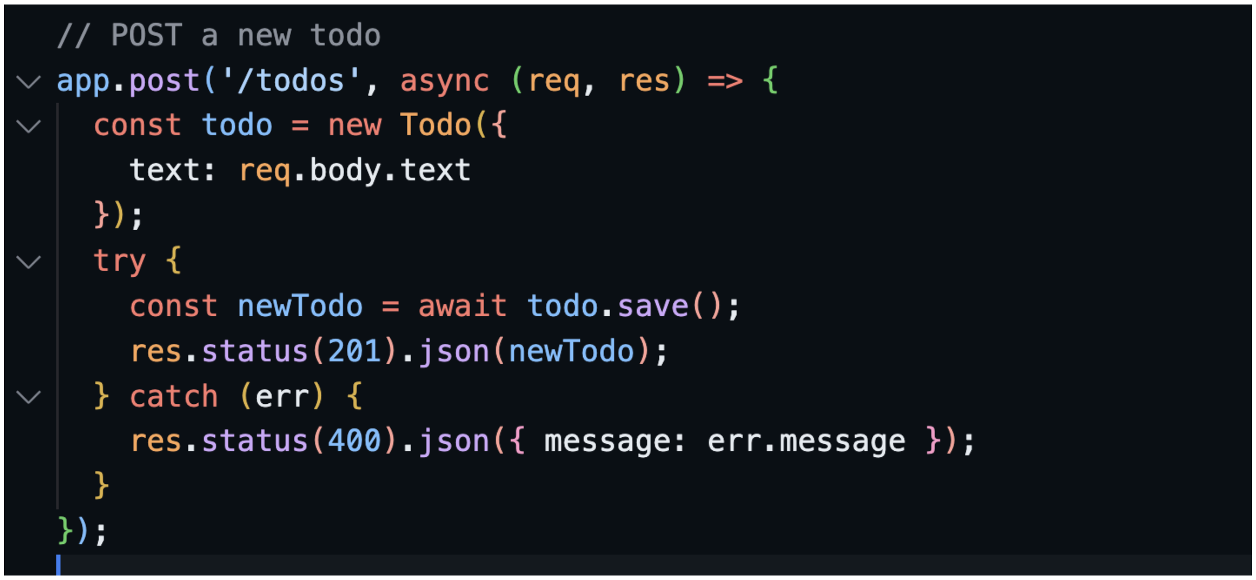


```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');

const app = express();
const PORT = 3000;

// Middleware
app.use(express.json());
app.use(cors());

// MongoDB Connection
mongoose.connect('mongodb://localhost:27017/todo-app-db', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
})
  .then(() => console.log('MongoDB connected successfully'))
  .catch(err => console.error('MongoDB connection error:', err));
```

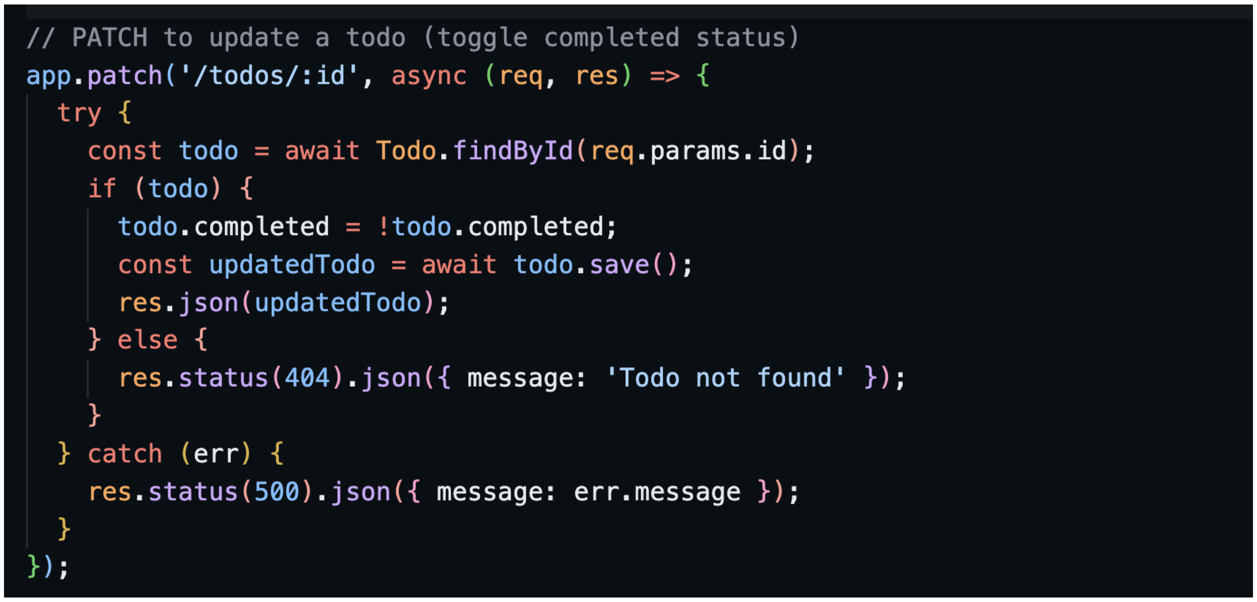
Curd operations



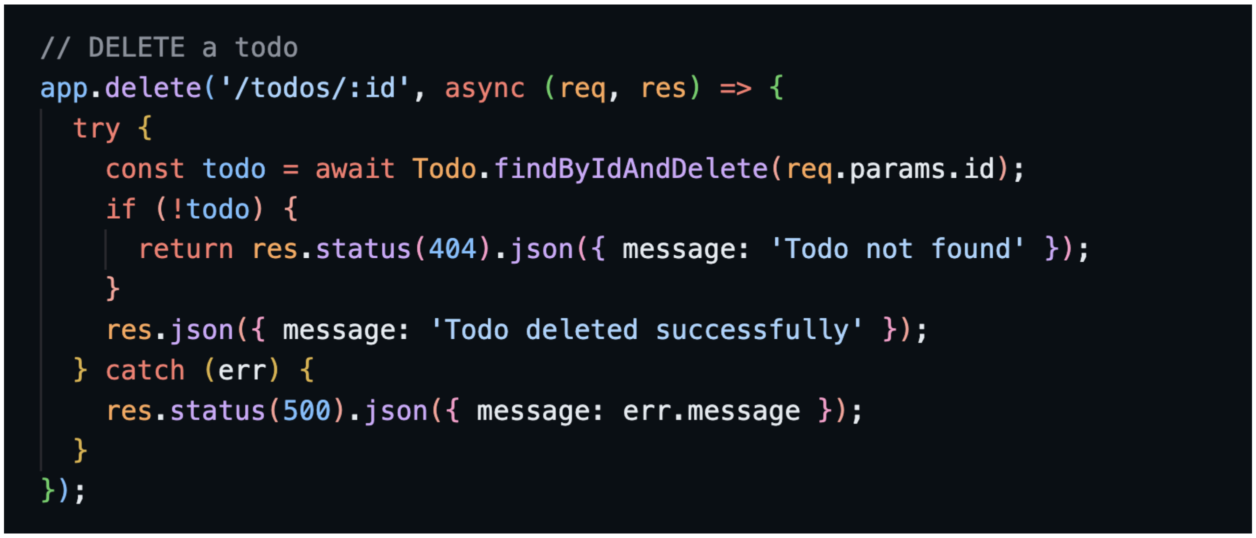
```
// GET all todos
app.get('/todos', async (req, res) => {
  try {
    const todos = await Todo.find().sort({ createdAt: -1 });
    res.json(todos);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
```



```
// POST a new todo
app.post('/todos', async (req, res) => {
  const todo = new Todo({
    text: req.body.text
  });
  try {
    const newTodo = await todo.save();
    res.status(201).json(newTodo);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});
```

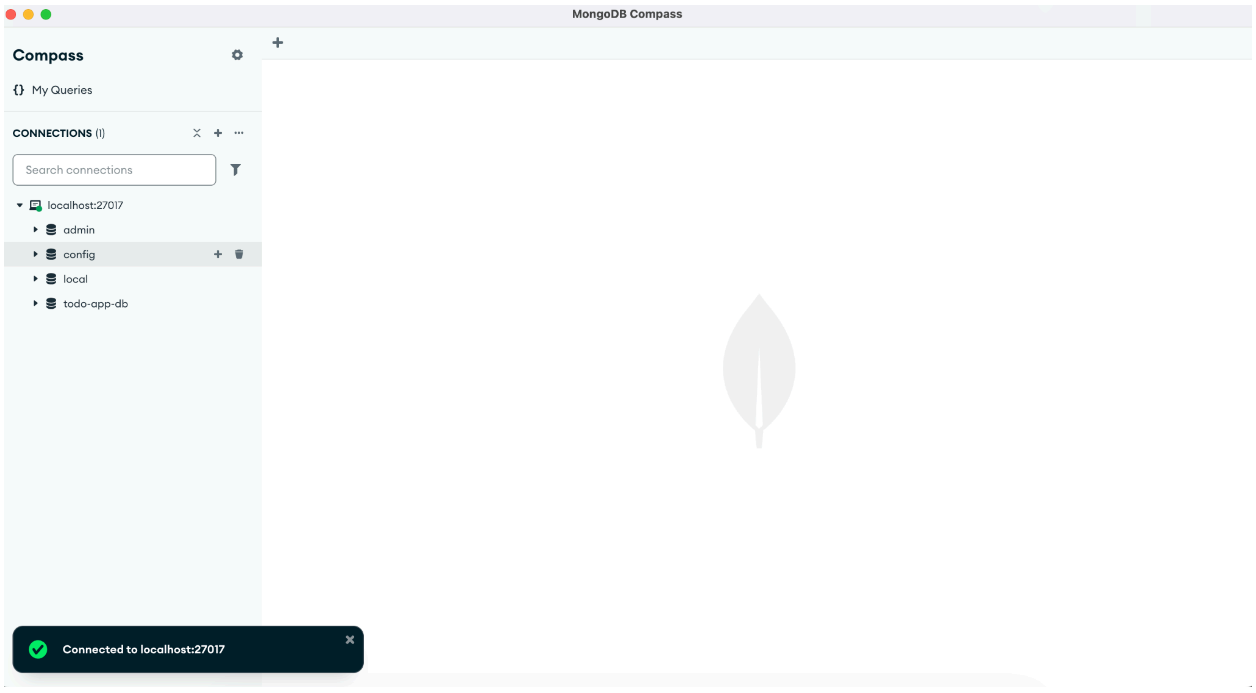


```javascript
// PATCH to update a todo (toggle completed status)
app.patch('/todos/:id', async (req, res) => {
  try {
    const todo = await Todo.findById(req.params.id);
    if (todo) {
      todo.completed = !todo.completed;
      const updatedTodo = await todo.save();
      res.json(updatedTodo);
    } else {
      res.status(404).json({ message: 'Todo not found' });
    }
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
```



```javascript
// DELETE a todo
app.delete('/todos/:id', async (req, res) => {
  try {
    const todo = await Todo.findByIdAndDelete(req.params.id);
    if (!todo) {
      return res.status(404).json({ message: 'Todo not found' });
    }
    res.json({ message: 'Todo deleted successfully' });
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
```

Output :


MongoDB Compass

Compass

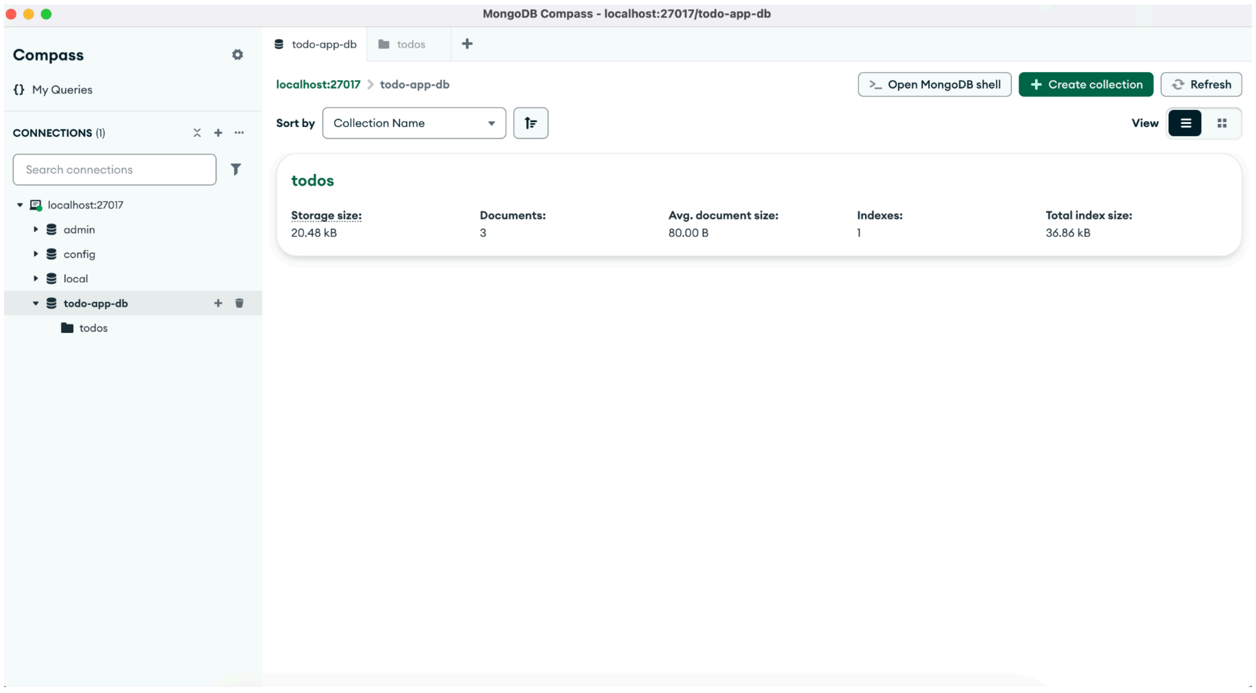
{} My Queries

CONNECTIONS (1)

Search connections

localhost:27017
admin
config
local
todo-app-db

Connected to localhost:27017


MongoDB Compass - localhost:27017/todo-app-db

Compass

{} My Queries

CONNECTIONS (1)

Search connections

localhost:27017
admin
config
local
todo-app-db
todos

todo-app-db | todos

localhost:27017 > todo-app-db

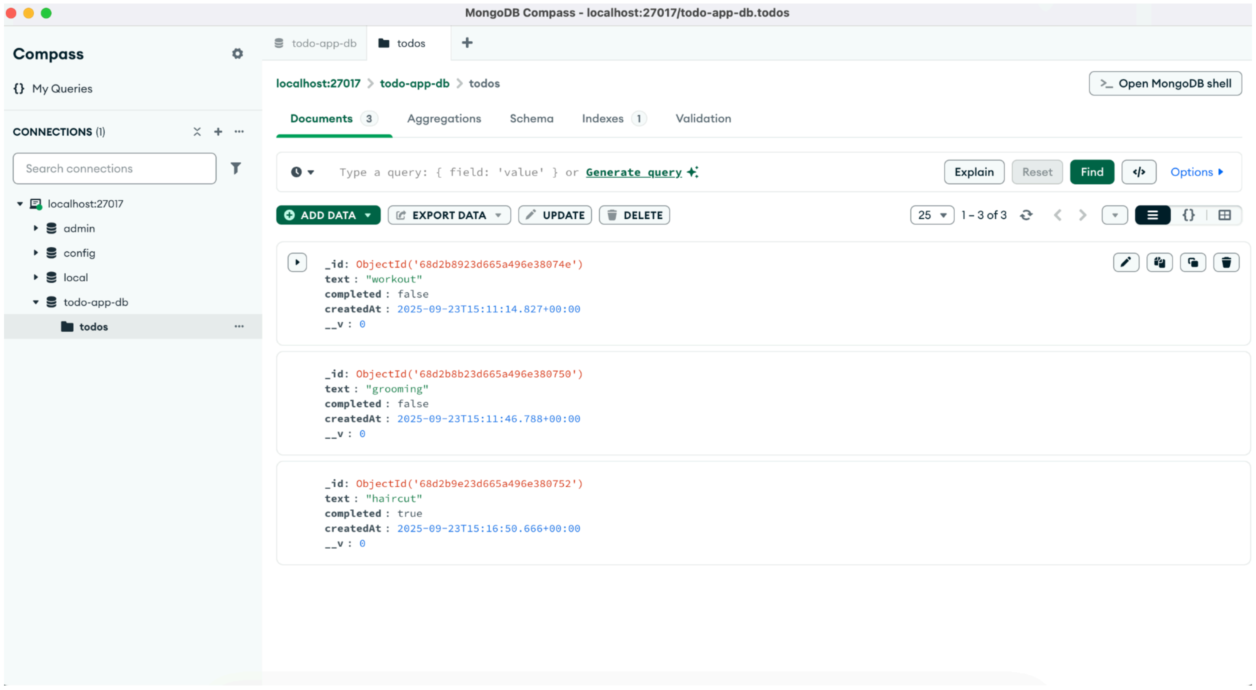
Open MongoDB shell | Create collection | Refresh

Sort by: Collection Name

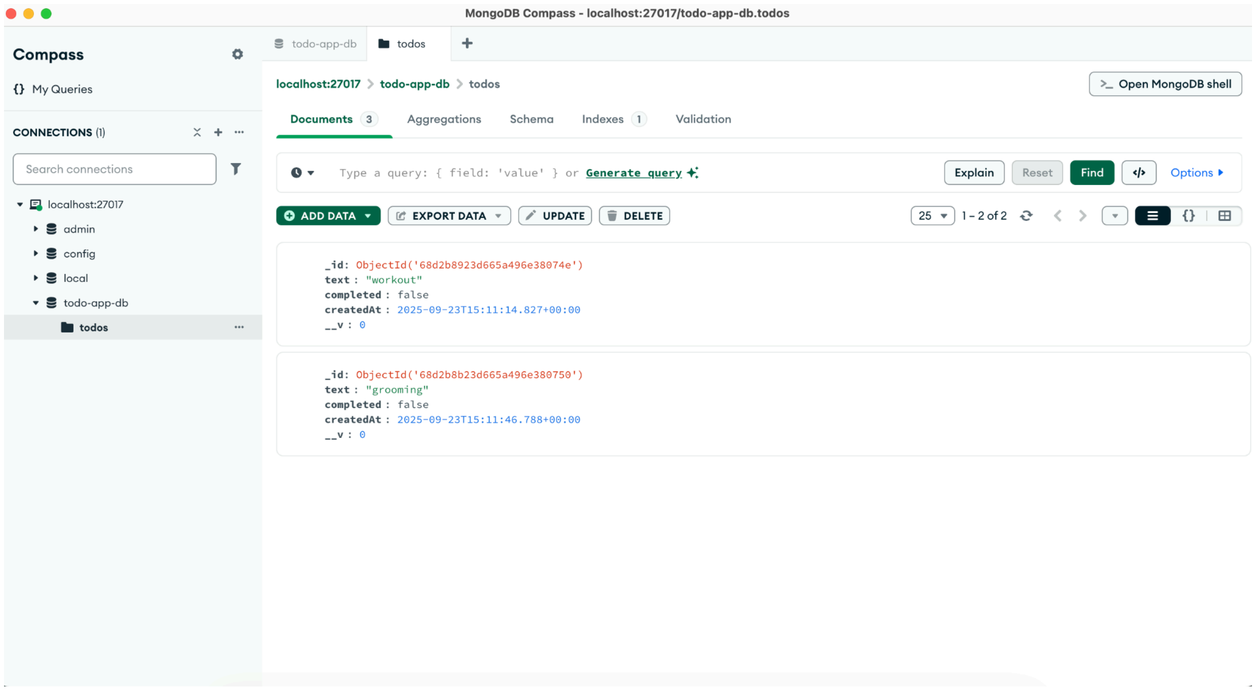
View

todos

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	3	80.00 B	1	36.86 kB



```
_id: ObjectId('68d2b8923d665a496e38074e')
text : "workout"
completed : false
createdAt : 2025-09-23T15:11:14.827+00:00
__v : 0

_id: ObjectId('68d2b8b23d665a496e380750')
text : "grooming"
completed : false
createdAt : 2025-09-23T15:11:46.788+00:00
__v : 0

_id: ObjectId('68d2b9e23d665a496e380752')
text : "haircut"
completed : true
createdAt : 2025-09-23T15:16:50.666+00:00
__v : 0
```



```
_id: ObjectId('68d2b8923d665a496e38074e')
text : "workout"
completed : false
createdAt : 2025-09-23T15:11:14.827+00:00
__v : 0

_id: ObjectId('68d2b8b23d665a496e380750')
text : "grooming"
completed : false
createdAt : 2025-09-23T15:11:46.788+00:00
__v : 0
```