Hướng Dẫn Triển Khai Case Study: Quản Lý To-Do List với MongoDB và Node.js

Tác giả: Đặng Kim Thi

1. Giới thiệu

Trong bài học này, chúng ta sẽ xây dựng một ứng dụng To-Do List cơ bản sử dụng Node.js và MongoDB. Ứng dụng này sẽ cho phép người dùng:

- Xem danh sách công việc.
- Thêm công việc mới.
- Cập nhật trạng thái hoàn thành của công việc.
- Xóa công việc.

Công nghệ sử dụng:

- **Node.js**: Môi trường runtime JavaScript.
- **MongoDB**: Cơ sở dữ liệu NoSQL để lưu trữ công việc.
- **Readline**: Giao diện dòng lệnh cho phép nhập/xuất từ người dùng.

---

2. Các bước triển khai

Bước 1: Cài đặt môi trường làm việc

1. **Cài đặt Node.js**: Tải và cài đặt Node.js từ nodejs.org.
2. **Cài đặt MongoDB**:
   - Cài đặt MongoDB từ mongodb.com.
   - Khởi động MongoDB trên cổng mặc định (`mongodb://localhost:27017`).
3. **Khởi tạo dự án Node.js**:

   ```
   mkdir todo-app
   cd todo-app
   npm init -y
   ```

   Lệnh này sẽ tạo ra file `package.json`.
4. **Cài đặt các thư viện cần thiết**:

   ```
   npm install mongodb readline
   ```

Bước 2: Kết nối tới MongoDB

1. **Import thư viện MongoDB**:

```
const { MongoClient } = require('mongodb');
```

2. **Tạo kết nối tới cơ sở dữ liệu**:

```
const uri = "mongodb://localhost:27017";
const client = new MongoClient(uri);
const dbName = "todoDB";
const collectionName = "tasks";
```

   - `uri`: Địa chỉ kết nối MongoDB.
   - `dbName`: Tên cơ sở dữ liệu.
   - `collectionName`: Tên bảng lưu trữ công việc.

3. **Mở kết nối**:

```
async function main() {
    try {
        await client.connect();
        console.log("Kết nối đến MongoDB thành công!");
    } catch (err) {
        console.error("Lỗi khi kết nối MongoDB:", err.message);
        process.exit(1);
    }
}
main();
```

Bước 3: Khởi tạo dữ liệu mẫu

1. **Mô tả dữ liệu mẫu**:

```
const sampleTasks = [
    { id: 1, description: "Hoàn thành báo cáo công việc", done: false
},
    { id: 2, description: "Lên kế hoạch họp nhóm", done: false },
    // Các công việc khác...
];
```

2. **Hàm khởi tạo dữ liệu**:

```
async function initializeSampleData() {
    const db = client.db(dbName);
```

```
    const collection = db.collection(collectionName);
    const existingTasks = await collection.countDocuments();

    if (existingTasks === 0) {
        console.log("Không có dữ liệu, khởi tạo 10 công việc mẫu...");
        await collection.insertMany(sampleTasks);
    } else {
        console.log("Dữ liệu đã tồn tại.");
    }
}
```

3. **Gọi hàm trong `main()`**:

```
await initializeSampleData();
```

Bước 4: Hiển thị menu và xử lý lựa chọn

1. **Tạo giao diện dòng lệnh**:

```
const readline = require('readline');
const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout
});
```

2. **Hiển thị menu**:

```
async function showMenu() {
    console.log("\n=== To-Do List ===");
    console.log("1. Xem tất cả công việc");
    console.log("2. Thêm công việc mới");
    console.log("3. Cập nhật công việc");
    console.log("4. Xóa công việc");
    console.log("5. Thoát");
    rl.question("Chọn một tùy chọn: ", handleOption);
}
```

3. **Xử lý lựa chọn**:

```
async function handleOption(option) {
    switch (option.trim()) {
        case '1':
            await viewTasks();
            break;
```

```
            case '2':
                await addTask();
                break;
            case '3':
                await updateTask();
                break;
            case '4':
                await deleteTask();
                break;
            case '5':
                console.log("Thoát chương trình. Tạm biệt!");
                rl.close();
                await client.close();
                process.exit(0);
            default:
                console.log("Lựa chọn không hợp lệ. Vui lòng thử lại.");
                break;
        }
        showMenu();
    }
```

Bước 5: Xem, thêm, cập nhật và xóa công việc

a) Xem tất cả công việc

```
async function viewTasks() {
    const db = client.db(dbName);
    const tasks = await db.collection(collectionName).find().toArray();
    console.log("\n=== Danh sách công việc ===");
    if (tasks.length === 0) {
        console.log("Không có công việc nào.");
    } else {
        tasks.forEach(task => {
            console.log(`[ID: ${task.id}] ${task.done ? "[X]" : "[ ]"}
${task.description}`);
        });
    }
}
```

b) Thêm công việc mới

```
async function addTask() {
    rl.question("Nhập mô tả công việc: ", async (description) => {
        const db = client.db(dbName);
        const newTask = { id: Date.now(), description, done: false };
        await db.collection(collectionName).insertOne(newTask);
        console.log("Công việc đã được thêm.");
```

```
        showMenu();
    });
}
```

c) Cập nhật trạng thái công việc

```
async function updateTask() {
    rl.question("Nhập ID công việc cần cập nhật: ", async (id) => {
        const taskId = parseInt(id.trim());
        const db = client.db(dbName);
        const task = await db.collection(collectionName).findOne({ id:
taskId });
        if (!task) {
            console.log("Không tìm thấy công việc.");
        } else {
            rl.question("Công việc đã hoàn thành chưa? (yes/no): ", async
(done) => {
                const isDone = done.trim().toLowerCase() === 'yes';
                await db.collection(collectionName).updateOne({ id: taskId
}, { $set: { done: isDone } });
                console.log("Công việc đã được cập nhật.");
                showMenu();
            });
        }
    });
}
```

d) Xóa công việc

```
async function deleteTask() {
    rl.question("Nhập ID công việc cần xóa: ", async (id) => {
        const taskId = parseInt(id.trim());
        const db = client.db(dbName);
        const result = await db.collection(collectionName).deleteOne({ id:
taskId });
        if (result.deletedCount === 0) {
            console.log("Không tìm thấy công việc.");
        } else {
            console.log("Công việc đã được xóa.");
        }
        showMenu();
    });
}
```

Bước 6: Chạy ứng dụng

1. **Chạy chương trình**:

```
node app.js
```

2. **Tương tác với menu để thực hiện các thao tác trên To-Do List.**

Lưu ý

- Đảm bảo MongoDB đang chạy trước khi khởi động ứng dụng.
- Sao lưu dữ liệu nếu cần thiết.

Chúc các em học tốt và zui zẻ 😉 Cô Thi

Phụ lục: Full code

```
const { MongoClient } = require('mongodb');
const readline = require('readline');

const uri = "mongodb://localhost:27017";
const client = new MongoClient(uri);
const dbName = "todoDB";
const collectionName = "tasks";

const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout
});

let nextId = 1;

const sampleTasks = [
    { id: nextId++, description: "Hoàn thành báo cáo công việc", done:
false },
    { id: nextId++, description: "Lên kế hoạch họp nhóm", done: false },
    { id: nextId++, description: "Dọn dẹp nhà cửa", done: true },
    { id: nextId++, description: "Đi mua sắm đồ ăn", done: false },
    { id: nextId++, description: "Tập thể dục buổi sáng", done: true },
    { id: nextId++, description: "Đọc sách về lập trình", done: false },
    { id: nextId++, description: "Chuẩn bị tài liệu thuyết trình", done:
false },
    { id: nextId++, description: "Học thêm một kỹ năng mới", done: false
},
    { id: nextId++, description: "Gọi điện cho gia đình", done: true },
    { id: nextId++, description: "Đi khám sức khỏe định kỳ", done: false }
];

async function initializeSampleData() {
```

```
    const db = client.db(dbName);
    const collection = db.collection(collectionName);
    const existingTasks = await collection.countDocuments();

    if (existingTasks === 0) {
        console.log("Không có dữ liệu, khởi tạo 10 công việc mẫu...");
        await collection.insertMany(sampleTasks);
    } else {
        console.log("Dữ liệu đã tồn tại.");
    }
}

async function showMenu() {
    console.log("\n=== To-Do List ===");
    console.log("1. Xem tất cả công việc");
    console.log("2. Thêm công việc mới");
    console.log("3. Cập nhật công việc");
    console.log("4. Xóa công việc");
    console.log("5. Thoát");
    rl.question("Chọn một tùy chọn: ", handleOption);
}

async function handleOption(option) {
    switch (option.trim()) {
        case '1':
            await viewTasks();
            break;
        case '2':
            await addTask();
            break;
        case '3':
            await updateTask();
            break;
        case '4':
            await deleteTask();
            break;
        case '5':
            console.log("Thoát chương trình. Tạm biệt!");
            rl.close();
            await client.close();
            process.exit(0);
        default:
            console.log("Lựa chọn không hợp lệ. Vui lòng thử lại.");
            break;
    }
    showMenu();
}

async function viewTasks() {
    const db = client.db(dbName);
    const tasks = await db.collection(collectionName).find().toArray();
    console.log("\n=== Danh sách công việc ===");
    if (tasks.length === 0) {
        console.log("Không có công việc nào.");
```

```
    } else {
        tasks.forEach(task => {
            console.log(`[ID: ${task.id}] ${task.done ? "[X]" : "[ ]"}
${task.description}`);
        });
    }
}

async function addTask() {
    rl.question("Nhập mô tả công việc: ", async (description) => {
        const db = client.db(dbName);
        const newTask = { id: nextId++, description, done: false };
        await db.collection(collectionName).insertOne(newTask);
        console.log("Công việc đã được thêm.");
        showMenu();
    });
}

async function updateTask() {
    rl.question("Nhập ID công việc cần cập nhật: ", async (id) => {
        const taskId = parseInt(id.trim());
        const db = client.db(dbName);
        const task = await db.collection(collectionName).findOne({ id:
taskId });
        if (!task) {
            console.log("Không tìm thấy công việc.");
        } else {
            rl.question("Công việc đã hoàn thành chưa? (yes/no): ", async
(done) => {
                const isDone = done.trim().toLowerCase() === 'yes';
                await db.collection(collectionName).updateOne({ id: taskId
}, { $set: { done: isDone } });
                console.log("Công việc đã được cập nhật.");
                showMenu();
            });
        }
    });
}

async function deleteTask() {
    rl.question("Nhập ID công việc cần xóa: ", async (id) => {
        const taskId = parseInt(id.trim());
        const db = client.db(dbName);
        const result = await db.collection(collectionName).deleteOne({ id:
taskId });
        if (result.deletedCount === 0) {
            console.log("Không tìm thấy công việc.");
        } else {
            console.log("Công việc đã được xóa.");
        }
        showMenu();
    });
}
```

```
async function main() {
    try {
        await client.connect();
        console.log("Kết nối đến MongoDB thành công!");
        await initializeSampleData();
        showMenu();
    } catch (err) {
        console.error("Lỗi khi kết nối MongoDB:", err.message);
        rl.close();
        process.exit(1);
    }
}

main();
```