แน่นอนครับ! ต่อไปนี้เป็นตัวอย่างเว็บแอปพลิเคชันที่ใช้ HTML, CSS, Node.js, และ SQL (จำลอง) ในการโหลดข้อมูลจำนวนมากจากฐานข้อมูลและแสดงผลในเว็บไซต์:

1. \*\*index.html\*\*: ไฟล์ HTML ที่จะโหลดลงในเบราว์เซอร์ของผู้ใช้

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>User Data</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h1>User Data</h1>

<div id="userData"></div>

<script src="script.js"></script>

</body>

</html>

```

2. \*\*styles.css\*\*: ไฟล์ CSS สำหรับการจัดรูปแบบเว็บไซต์

```css

body {

font-family: Arial, sans-serif;

}

h1 {

text-align: center;

}

#userData {

margin: 0 auto;

width: 80%;

}

```

3. \*\*script.js\*\*: ไฟล์ JavaScript สำหรับโหลดข้อมูลจาก server-side และแสดงผลในเว็บ

```javascript

window.onload = function() {

fetch('/users')

.then(response => response.json())

.then(data => {

const userDataDiv = document.getElementById('userData');

data.forEach(user => {

const userElement = document.createElement('div');

userElement.innerHTML = `<p>Name: ${user.name}</p><p>Email: ${user.email}</p>`;

userDataDiv.appendChild(userElement);

});

})

.catch(error => console.error('Error fetching user data:', error));

};

```

4. \*\*server.js\*\*: ไฟล์ Node.js ที่จะทำหน้าที่เป็นเซิร์ฟเวอร์และจัดการข้อมูลจากฐานข้อมูล

```javascript

const http = require('http');

const fs = require('fs');

const sqlite3 = require('sqlite3').verbose();

// เปิดฐานข้อมูล SQLite

const db = new sqlite3.Database(':memory:');

// สร้างตาราง users และเพิ่มข้อมูล

db.serialize(() => {

db.run('CREATE TABLE users (id INTEGER PRIMARY KEY, name TEXT, email TEXT)');

const stmt = db.prepare('INSERT INTO users (name, email) VALUES (?, ?)');

for (let i = 0; i < 100000; i++) {

stmt.run(`User ${i}`, `user${i}@example.com`);

}

stmt.finalize();

});

// สร้างเซิร์ฟเวอร์

const server = http.createServer((req, res) => {

if (req.url === '/users') {

// อ่านข้อมูลผู้ใช้จากฐานข้อมูล

db.all('SELECT \* FROM users', (err, rows) => {

if (err) {

console.error(err);

res.writeHead(500, {'Content-Type': 'text/plain'});

res.end('Internal Server Error');

} else {

res.writeHead(200, {'Content-Type': 'application/json'});

res.end(JSON.stringify(rows));

}

});

} else {

// อ่านไฟล์ index.html

fs.readFile('index.html', (err, data) => {

if (err) {

console.error(err);

res.writeHead(500, {'Content-Type': 'text/plain'});

res.end('Internal Server Error');

} else {

res.writeHead(200, {'Content-Type': 'text/html'});

res.end(data);

}

});

}

});

// กำหนดพอร์ตที่เซิร์ฟเวอร์จะใช้

const PORT = process.env.PORT || 3000;

// เริ่มเซิร์ฟเวอร์และรอการเชื่อมต่อ

server.listen(PORT, () => {

console.log(`Server is running on port ${PORT}`);

});

```

โค้ดนี้จะสร้างเซิร์ฟเวอร์ Node.js ที่จะสร้างฐานข้อมูล SQLite ในหน่วยความจำ (in-memory) และเพิ่มข้อมูลผู้ใช้ 100,000 รายการ โดยเมื่อผู้ใช้เข้าชมเว็บไซต์ที่จะโหลดข้อมูลผู้ใช้จากฐานข้อมูล ซึ่งจะแสดงผลทั้งหมดในหน้าเว็บไซต์

จะเห็นว่าโค้ดนี้เป็นตัวอย่างขนาดใหญ่ แต่มันเป็นวิธีการแบ่งงานระหว่าง client-side และ server-side ที่มีประสิทธิภาพและทำงานได้อย่างมีประสิทธิภาพในสถานการณ์ที่มีการโหลดข้อมูลจำนวนมากจากฐานข้อมูล หวังว่าตัวอย่างนี้จะเป็นประโยชน์ครับ!