УДК 004.6

ІНТЕГРАЦІЯ MONGODB ТА NODE.JS: СУЧАСНИЙ ПІДХІД ДО РОЗРОБКИ ВЕБ-ДОДАТКІВ ДЛЯ ОПТИМІЗАЦІЇ УПРАВЛІННЯ РЕСУРСАМИ ПРОМИСЛОВОЇ КОМПАНІЇ

Олінкевич Я.В., Колесник Л.В.

Харківський національний університет радіоелектроніки, Харків

У світі сучасної веб-розробки продуктивна робота та швидкодія баз даних та серверних мов програмування є ключовими чинниками успіху будь-якого застосунку. Особливо важливими ці фактори стають тоді, коли виникає необхідність створення ефективного та багатофункціонального Інтернет-додатку для задоволення потреб промислової компанії. Отже, виникає питання вдалого вибору програмних інструментів для реалізації даної задачі серед безлічі існуючих.

На даний момент доволі популярними технологічними стеками для веб-розробки є Node.js та MongoDB. Node.js, потужна серверна платформа, що працює на движку V8 від Google, дозволяє легко створювати швидкі та гнучкі додатки, в той час як MongoDB ідеально підходить в якості масштабованої, високопродуктивної NoSQL бази даних з відкритим вихідним кодом [1]. Але як саме інтеграція MongoDB та Node.js може трансформувати розробку додатків, розкриваючи потенціал обох технологій, та допомогти створити гідний корпоративний застосунок?

Передова мова JavaScript стала рушійною силою розвитку Node.js, потужної платформи, яка дозволяє використовувати JavaScript на стороні сервера [2]. Завдяки своїй асинхронній, подієво-орієнтованій архітектурі та широкій підтримці з боку розробників, Node.js стала популярним вибором для створення швидких та ефективних веб-додатків. Використання JavaScript як єдиної мови на клієнтській та серверній сторонах значно спрощує розробку, оскільки розробники можуть легко перемикатися між фронтендом та бекендом. Крім того, величезна кількість модулів, доступних через систему управління пакетами Node (NPM), дозволяє розширювати функціональність додатків.

MongoDB – це провідна NoSQL база даних, яка використовується для сучасних

додатків, що потребують швидкого доступу та гнучкості у роботі з великими обсягами даних. Основною одиницею даних у MongoDB є «документ», що дозволяє зберігати дані у форматі, схожому на JSON, відомому як BSON. Це надає розробникам змогу легко маніпулювати даними, використовуючи добре знайомі JavaScript об'єкти. У контексті веб-розробки, де швидкість та масштабованість є критично важливими, MongoDB пропонує можливості, які традиційні реляційні бази даних не можуть надати. Її гнучка схема документів дозволяє додавати, видаляти та змінювати поля без потреби переконфігурувати всю базу даних.

Однією з ключових переваг інтеграції MongoDB з Node.js є їх спільна основа даних – JSON. MongoDB зберігає дані у форматі BSON (рис. 1), який є бінарною репрезентацією JSON. Така уніфікація дозволяє розробникам з легкістю передавати дані між базою даних та сервером, зменшуючи необхідність конвертації або серіалізації даних. Офіційний драйвер MongoDB Node.js робить роботу з MongoDB всередині скрипта Node.js простою та інтуїтивно зрозумілою для розробників, заощаджуючи час та підвищуючи продуктивність [3]. Він відіграє центральну роль у цій інтеграції, оскільки забезпечує програмний інтерфейс для взаємодії з базою даних. Ця бібліотека дозволяє розробникам використовувати знайомі конструкції JavaScript для виконання операцій з даними.

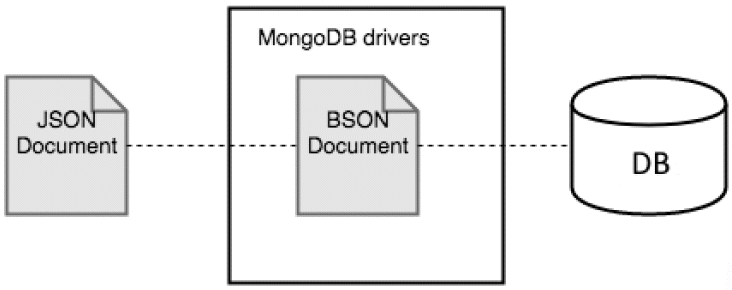

Рисунок 1 – Конвертація JSON у BSON та зберігання у БД

На рис. 2 розглянемо приклад роботи з базою даних MongoDB [4].

```
var MongoClient = require('mongodb').MongoClient;
// Підключення до бази даних
MongoClient.connect("mongodb://localhost:27017/exampleDb",
function(err, db) { if(err) { return console.error(err); }
// Звернення або створення колекції widgets
db.collection('widgets', function(err, collection) {
if (err) return console.error(err);
// Видалення всіх елементів
collection.remove(null,{safe : true}, function(err, result) {
if (err) return console.error(err);
console.log('result of remove ' + result.result);
// Створення двох записів
var widget1 = {title : 'First Great widget'
        desc : 'greatest widget of all',
price : 14.99};
var widget2 = {title : 'Second Great widget',
        desc : 'second greatest widget of all',
        price : 29.99};
collection.insertOne(widget1, {w:1}, function (err, result) {
        if (err) return console.error(err);
console.log(result.insertedId);
collection.insertOne(widget2, {w:1}, function(err, result) {
        if (err) return console.error(err);
        console.log(result.insertedId);
        collection.find({}).toArray(function(err,docs) {
        console.log('found documents');
console.dir(docs);
// Закриття бази даних
db.close(); });});});});});});});
```

Рисунок 2 – Приклад роботи з MongoDB

За допомогою класу MongoClient, легко здійснюється підключення до локальної інстанції MongoDB і виконання ряду операцій з колекцією widgets. Цей код відображає типовий потік роботи з базою даних: від підключення до неї, керування колекціями і документами, до видалення, вставки та вибірки даних, і, нарешті, закриття з'єднання.

Окрім стандартного драйвера, існують і фреймворки, які розширюють функціональність MongoDB і спрощують роботу з базою даних. Одним з найбільш популярних є Mongoose (рис. 3). Він пропонує схеми для моделювання даних, валідацію, побудову запитів та багато іншого, що дозволяє розробникам швидше реалізовувати складні додатки. Використання фреймворків на кшталт Mongoose може значно підвищити продуктивність розробки, надаючи додаткові можливості для роботи з даними, такі як автоматичне створення ідентифікаторів, простіша робота з відносинами між документами, та вбудовані механізми для підтримки складної бізнес-логіки в моделях даних.

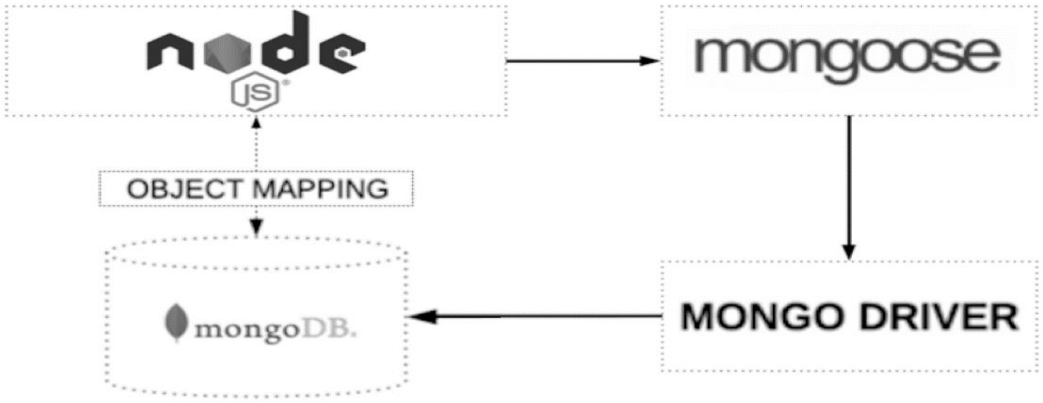

Рисунок 3 – Маппінг (зв'язування) об'єктів між Node та MongoDB, що управляється через Mongoose

У результаті дослідження було виявлено, що Node.js є актуальним та ефективним інструментом для розробки серверної частини веб-додатків, а MongoDB пропонує швидку роботу з даними завдяки своїй документо-орієнтованій структурі. Їх інтеграція пропонує гнучкий підхід до розробки, який спрощує передачу та обробку даних. Використання таких технологій є релевантним для сучасних веб-проектів, що вимагають високої продуктивності та масштабованості, і, як наслідок, є вкрай вдалим рішенням для розв'язання задачі оптимізації управління ресурсами промислового підприємства.

Література:

1. M. Satheesh , B. J. D`mello, and J. Krol, Web Development with MongoDB and NodeJS: Build an interactive and full-featured web application from scratch using Node.js and MongoDB. Birmingham, UK: Packt Publishing, 2015.

2. M. Pirtle, MongoDB for Web Development. London, UK: Pearson Education, 2018.

3. MongoDB With Node.js. MongoDB. [Он-лайн]. Доступно: https://www.mongodb.com/languages/mongodb-with-nodejs.

4. S. Powers, Learning Node: Moving to the Server-Side. Sebastopol, USA: O'Reilly Media, 2016.