**Why Node.js and MongoDB technology**

Performance and scalability

Node.js is built upon Google Chrome’s V8 runtime—written in C++, built for multiple operating systems and superfast. Both V8 and Node are frequently updated, with performance optimizations, security patches, and support for modern JavaScript features. The use of JavaScript also means that transforming JSON data—the most common data interchange format on the Web—is fast by default.

Combining Node.js with a Microservices

A microservice is a single self-contained unit which, together with many others, makes up a large application. By splitting your app into small units every part of it is independently deployable and scalable, can be written by different teams and in different programming languages and can be tested individually.

That also means you can run separate teams in Poland, India, and the US, and they can work independently on different parts of the same app. Smaller teams are much easier to work with, plus, you don’t have to deploy the whole codebase every time you want to introduce some change in one part of your app.

There’s a number of well-known enterprise-level companies who have already embraced the power of Node.js and microservices, a perfect combination for better performance, less code, and efficient deployment. PayPal, Netflix, and GoDaddy are just a handful of them.

Client and Server Side

As Node.js is JavaScript, it fills the gap between frontend and backend skills for developers, which means they can act as full-stack devs without having to learn additional languages

MongoDB

Document-based data storage is the main aim of using a non-structured database like NoSQL. MongoDB is a distributed database which allows ad-hoc queries, real-time integration, and indexing efficient. Moreover, MongoDB is open-source and perfect for frequently changing data. It also offers server-side data validation.

The data we talked about in the above points is generally semi-structured or unstructured data. If your application produces data in a format that doesn’t fit well in a table. As an example, a blog writing platform can store articles and its metadata in tables and retrieve one blog with a complex/nested/join query. On the other hand, store one entire blog as a single document.

All these platforms are not only needed to store data but also need to have high read/write performance on the same. Consider a chat application, which has thousands of messages being sent and received every second. The amount of data generated is huge, not-structured and requires high performance.

You are working on a project with rapid time-to-market, with development practices like agile sprints. NoSQL databases are easier and faster to develop software.

If your application is heavily dependent on transactions, it is recommended to use an RDBMS because they are more stable, support data integrity and has ACID properties (Note: In a recent release, MongoDB now also supports ACID natively).