MERN Stack Explained

MERN is one of several variations of the MEAN stack (MongoDB Express Angular Node), where the traditional Angular.js front-end framework is replaced with React.js. Other variants include MEVN (MongoDB, Express, Vue, Node), and really any front-end JavaScript framework can work.

Ready to take the next step? Set up your free Atlas account by clicking below and try our MERN Stack Tutorial to create a full-stack MERN application in no time.

Get started free

What is the MERN stack?

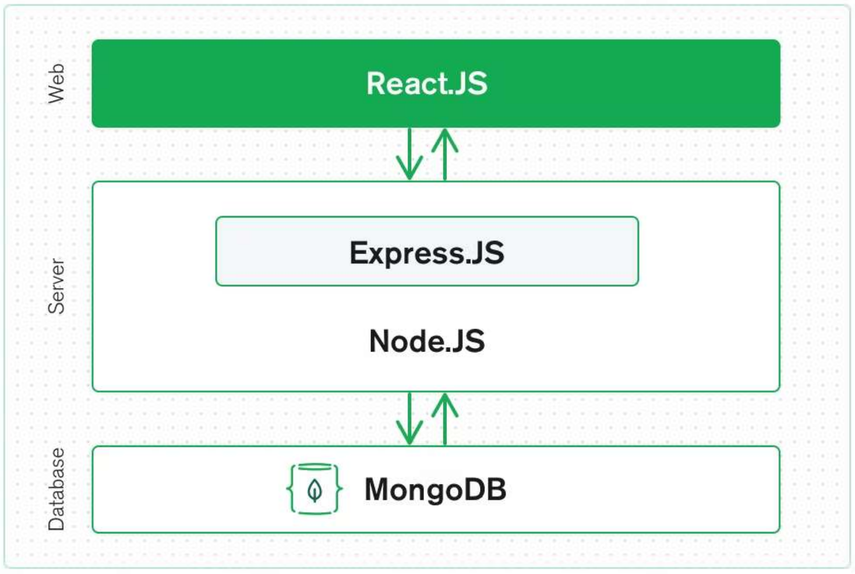
MERN stands for MongoDB, Express, React, Node, after the four key technologies that make up the stack.

- MongoDB — document database
- Express(.js) — Node.js web framework
- React(.js) — a client-side JavaScript framework
- Node(.js) — the premier JavaScript web server

Express and Node make up the middle (application) tier. Express.js is a server-side web framework, and Node.js is the popular and powerful JavaScript server platform. Regardless of which variant you choose, ME(RVA)N is the ideal approach to working with JavaScript and JSON, all the way through.

How does the MERN stack work?

The MERN architecture allows you to easily construct a three-tier architecture (front end, back end, database) entirely using JavaScript and JSON.



React.js front end

The top tier of the MERN stack is React.js, the declarative JavaScript framework for creating dynamic client-side applications in HTML. React lets you build up complex interfaces through simple components, connect them to data on your back-end server, and render them as HTML.

React's strong suit is handling stateful, data-driven interfaces with minimal code and minimal pain, and it has all the bells and whistles you'd expect from a modern web framework: great support for forms, error handling, events, lists, and more.

Express.js and Node.js server tier

The next level down is the Express.js server-side framework, running inside a Node.js server. Express.js bills itself as a "fast, unopinionated, minimalist web framework for Node.js," and that is indeed exactly what it is. Express.js has powerful models for URL routing (matching an incoming URL with a server function), and handling HTTP requests and responses.

By making XML HTTP Requests (XHRs) or GETs or POSTs from your React.js front end, you can connect to Express.js functions that power your application. Those functions, in turn, use MongoDB's Node.js drivers, either via callbacks or using promises, to access and update data in your MongoDB database.

MongoDB database tier

If your application stores any data (user profiles, content, comments, uploads, events, etc.), then you're going to want a database that's just as easy to work with as React, Express, and Node.

That's where MongoDB comes in: JSON documents created in your React.js front end can be sent to the Express.js server, where they can be processed and (assuming they're valid) stored directly in MongoDB for later retrieval. Again, if you're building in the cloud, you'll want to look at Atlas. If you're looking to set up your own MERN stack, read on!

Is MERN a full-stack solution?

Yes, MERN is a full stack, following the traditional three-tier architectural pattern, including the front-end display tier (React.js), application tier (Express.js and Node.js), and database tier (MongoDB).

Why choose the MERN stack?

Let's start with MongoDB, the document database at the root of the MERN stack. MongoDB was designed to store JSON data natively (it technically uses a binary version of JSON called BSON), and everything from its command line interface to its query language (MQL, or MongoDB Query Language) is built on JSON and JavaScript.

MongoDB works extremely well with Node.js, and makes storing, manipulating, and representing JSON data at every tier of your application incredibly easy. For cloud-native applications, MongoDB Atlas makes it even easier, by giving you an auto-scaling MongoDB cluster on the cloud provider of your choice, as easy as a few button clicks.

Express.js (running on Node.js) and React.js make the JavaScript/JSON application MERN full stack, well, full. Express.js is a server-side application framework that wraps HTTP requests and responses, and makes it easy to map URLs to server-side functions. React.js is a front end JavaScript framework for building interactive user interfaces in HTML, and communicating with a remote server.

The combination means that JSON data flows naturally from front to back, making it fast to build on and reasonably simple to debug. Plus, you only have to know one programming language, and the JSON document structure, to understand the whole system!

MERN is the stack of choice for today's web developers looking to move quickly, particularly for those with React.js experience.

MERN use cases

Like any web stack, you can build whatever you want in MERN — though it's ideally suited for cases that are JSON-heavy, cloud-native, and that have dynamic web interfaces.

Examples include workflow management, news aggregation, to-do apps and calendars, and interactive forums/social products — and whatever else you can dream up!

Related resources

- **Take the next step**: try our MERN Stack Tutorial to create a full-stack MERN application in no time
- What Is the MEAN Stack?
- What Is a Database-as-a-Service?
- Beginners Guide: MongoDB Basics

Get Started Free

MongoDB.

English

About

Careers

Investor Relations

Legal Notices

Privacy Notices

Security Information

Trust Center

Support

Contact Us

Customer Portal

Atlas Status

Paid Support

Social

Github

Stack Overflow

LinkedIn

Youtube

Twitter

Twitch

Facebook