**How to choose the right tech stack for web development -** [The Educative Team](https://educative-inc.medium.com/?source=post_page-----31640a49754c--------------------------------) Jul 28, 2022

There are a lot of different technology stacks for web development out there. Knowing what is available and when to use them can be daunting, especially for beginners. In this article, we will explore some of the most popular tech stacks for web development and look at some of the pros and cons of each.

Afterward, we’ll give you some tips on how to choose the right tech stack for your project, whether it’s web development, mobile app development or software development in general. We’ll conclude with resources and further reading if you want to dive deeper into the subject.

Let’s get started!

**We’ll cover**:

* What is a tech stack?
* **Popular tech stacks for web development**
* 1. LAMP
* *WAMP & MAMP*
* 2. MEAN
* *MERN & MEVN*
* 3. Django
* Which stack should you use?
* Tips for picking and mastering a stack
* Wrapping up and next steps

**What is a tech stack?**

First, what is a technology stack?

A tech stack is a combination of programming languages, databases, and frameworks used to create a web application or website.

A typical web development stack is usually a mix of front-end and back-end technologies that include:

* **Framework**: Libraries of code written by other developers. These can help you build a web application without starting from scratch.
* **Web server/HTTP servers**: The HTTP (Hypertext Transfer Protocol) server handles requests such as sending or receiving emails, downloading files, etc.
* **Databases**: A database stores and organizes data. The data can be retrieved, manipulated, updated, and managed in a variety of different ways.
* **Programming languages**: These are used to communicate instructions in a way that is understandable by computers.
* [**Operating systems (OS)**](https://www.educative.io/blog/operating-systems-crashcourse?eid=5082902844932096): Software that manages the hardware, software, and other applications or resources of a computer.

Multiple technology stacks exist for every development need, and the right stack will depend on your project requirements. For example, if you need a high-performance website or web application that can handle a lot of traffic and data, you would need a tech stack with robust back-end support.

Similarly, if you want to build a website that needs to be up and running quickly with limited functionality and resources (such as a simple landing page), you will need a tech stack that is lightweight and easy to set up.

**Popular tech stacks for web development**

There are many different tech stacks that can be used for web development, but in this article, we’ll cover:

* **LAMP** stack: Linux, Apache, MySQL, and PHP/Perl/Python
* **WAMP** stack: Windows, Apache, MySQL, and PHP/Perl/Python
* **MAMP** stack: macOS, Apache, MySQL, and PHP/Perl/Python
* **MEAN** stack: MongoDB, Express.js, AngularJS, and Node.js
* **MERN** stack: MongoDB, Express.js, ReactJS, and Node.js
* **MEVN** stack: MongoDB, Express.js, Vue.js, and Node.js
* **Django**: Django, Python

**What to consider when choosing a tech stack**

* **Define your needs**: What will your product do, and what does it require to fulfill your vision?
* **Research your market**: Investigate potential competitors, seek out unmet needs as opportunities, and analyze your target audience.
* **Determine your product’s viability**: Create a prototype (aka minimum viable product or MVP) and test it with your target market to see if it has a use.
* **Think about scalability**: As your product grows, how will you accommodate more users without losing quality or control? What do different technology stacks offer as solutions for scaling?
* **Consider security**: How will you protect your web application and user data?
* **Compare costs**: Some technology stacks will require more expensive hosting or other resources than others. Choose the one that makes the most financial sense for your product.

**LAMP**

LAMP is a classic technology stack that powers several high-traffic websites like Facebook, Wikipedia, and Tumblr.

The LAMP stack consists of:

* **Linux** (OS)
* **Apache** (server)
* **MySQL** (database)
* **PHP/Perl/Python** (programming language)

**Linux**: The Linux OS forms the foundation of the LAMP stack. It’s an [open-source](https://www.educative.io/blog/what-is-open-source-software-guide?eid=5082902844932096) platform that’s popular among developers because it’s free and easily customizable. In comparison, Windows is a closed-source platform, so you don’t have as much control over how it works. Linux also enjoys a large community of developers and extensive documentation.

**Apache**: Apache is a cross-platform, open-source HTTP server that can handle large amounts of traffic and data. It’s fast, secure, and reliable, having been around since 1996. Apache is modular and can be customized to optimize its performance as needed.

**MySQL**: MySQL is a relational database management system (RDBMS) which means that it stores and presents data in tabular form, organized in rows and columns. It is open-source and easy to use. MySQL is scalable and can handle tremendous quantities of data, but its performance and efficiency can suffer when dealing with exceptionally large databases.

**PHP/Perl/Python**: PHP is the most popular programming language used with the LAMP stack, and is compatible with virtually all servers. It’s relatively easy to learn and has a wide range of libraries and frameworks available. Perl and Python are also sometimes used in place of PHP.

* **Perl**: A powerful, high-level, object-oriented language with a complex syntax that makes it more difficult for beginners to pick up.
* **Python**: A versatile, highly-extensible language popular among developers for its readability, comprehensibility, and ease of use. Python is also a popular language used in scientific and numeric computing.

Graphical user interface, application, table

Description automatically generated

**WAMP and MAMP**

WAMP and MAMP are variants of the LAMP stack that are designed to be used on Microsoft Windows and macOS, respectively.

**MEAN**

The MEAN stack is a JavaScript web framework that has been used to create websites like YouTube, Netflix, and PayPal. MEAN is a great tech stack for people who are already familiar with JavaScript or want to speed up their development process.

The MEAN stack consists of:

* **MongoDB** (database)
* **Express.js** (web framework for Node.js)
* **Angular.js** (framework)
* **Node.js** (server)

[**MongoDB**](https://www.educative.io/blog/mongodb-vs-mysql?eid=5082902844932096): MongoDB is an open-source database that stores JSON documents created by the front-end framework (in this case, Angular.js) after they have been processed by Express.js. MongoDB can store a massive amount of data, facilitate fast data exchanges between the client-side and server, and is cloud-compatible.

**Express.js**: Express is a lightweight back-end web framework that runs on top of the Node.js server. Express.js excels at URL routing and handling HTTP requests. Express.js receives JSON documents from the web framework and processes them so they can be stored in MongoDB.

[**Angular.js**](https://www.educative.io/blog/angularjs-interview-questions?eid=5082902844932096): Angular is speedy, open-source, and a full-fledged front-end framework. Angular.js is strongly (but not strictly) associated with the model-view-controller (MVC) architecture.

[**Node.js**](https://www.educative.io/blog/what-is-nodejs?eid=5082902844932096): Node.js is an asynchronous event-driven JavaScript runtime environment. As an asynchronous server, it’s very memory efficient. It is open-source, and cross-platform, running Windows, Linux, macOS, etc.).

Graphical user interface, application

Description automatically generated

**MERN and MEVN**

MERN and MEVN are both very similar to the MEAN stack, but use [React](https://www.educative.io/blog/what-is-react-tutorial?eid=5082902844932096) and Vue.js, respectively.

We’ll start off by looking at MERN, and then talk about what makes MEVN different.

**MERN**

The MERN stack is very similar to the MEAN and MEVN stacks but uses the React JavaScript library instead of Angular.js or Vue.js as its front-end framework. MERN is used at companies like Dropbox, Facebook, and Airbnb. MERN enjoys many of the same benefits as the MEAN stack, but React has a gentler learning curve than Angular.js.

The MERN stack consists of:

* **MongoDB** (database)
* **Express.js** (web framework for Node.js)
* **React** (front-end “framework”)
* **Node.js** (server)

**MongoDB**: MongoDB is an open-source database that stores JSON documents created by the front-end framework (in this case, React) after they have been processed by Express.js. MongoDB can store a massive amount of data, facilitate fast data exchanges between the client-side and server, and is cloud-compatible.

**Express.js**: Express is a lightweight back-end web framework that runs on top of the Node.js server. Express.js excels at URL routing and handling HTTP requests. Express.js receives JSON documents from the web framework and processes them so they can be stored in MongoDB.

**React**: A fast, scalable, open-source JavaScript library used as the front-end framework for MERN. React provides a virtual DOM that makes it easy to manipulate elements on a web page, but slightly slower than MEAN, which uses a native DOM. React also uses a component-based library, and this makes it easy to reuse code across different parts of your application. React is also really easy to learn if you already happen to know JavaScript.

**Node.js**: Node.js is an asynchronous event-driven JavaScript runtime environment. As an asynchronous server, it’s very memory efficient. It is open-source, and cross-platform, running Windows, Linux, macOS, etc.).

**MEVN**

MEVN uses [Vue.js](https://www.educative.io/m/vue-js-tutorial?eid=5082902844932096) as its front-end framework instead of Angular.js or React. Vue.js is a JavaScript framework for building user interfaces, and it’s one of the most popular frameworks out there. The MEVN stack has been used to build websites like Alibaba, Behance, and GitLab. It is also compatible with Meteor and Docker.

The MEVN stack consists of:

* **MongoDB** (database)
* **Express.js** (web framework for Node.js)
* **Vue.js** (front-end framework)
* **Node.js** (server)

**MongoDB**: MongoDB is an open-source database that stores JSON documents created by the front-end framework (in this case, React) after they have been processed by Express.js. MongoDB can store a massive amount of data, facilitate fast data exchanges between the client-side and server, and is cloud-compatible.

**Express.js**: Express is a lightweight back-end web framework that runs on top of the Node.js server. Express.js excels at URL routing and handling HTTP requests. Express.js receives JSON documents from the web framework and processes them so they can be stored in MongoDB.

**Vue.js**: Vue.js is a progressive framework based on the MVVM architecture. Adding new features or functions is a simple process (just add packages) and can be integrated into projects incrementally. Vue.js has an intuitive, easy-to-read syntax that makes it popular amongst developers in the open-source community.

**Node.js**: Node.js is an asynchronous event-driven JavaScript runtime environment. As an asynchronous server, it’s very memory efficient. It is open-source, and cross-platform, running Windows, Linux, macOS, etc.).

**Django**

Django is a high-level [Python web development](https://www.educative.io/blog/web-development-in-python?eid=5082902844932096) framework that encourages rapid development and clean, well-organized code. It’s popular among web developers because of its flexible architecture and ease of use, however, it can be hard to learn. The Django tech stack has been used to build websites like Instagram and Pinterest, although Pinterest has since moved to Flask.

The Django stack will always have Django and Python, but the server and database you choose can differ depending on your preference.

One example of a Django tech stack:

* **Django** (web framework)
* **Python** (programming language)
* **Apache** (server)
* **MySQL** (database)

**Django**: A comprehensive and versatile web framework with plenty of features and functionality straight out of the box. Django prioritizes **rapid** development and clean design. Django is compatible with a variety of web hosting providers and databases and provides an admin panel, ORM, templates, authentication (through package), and more.

**Python**: A versatile, highly-extensible language that is popular among developers for its readability, comprehensibility, and ease of use. Python is also used widely in machine learning applications, and in scientific and numeric computing.

**Apache**: Apache is a cross-platform, open-source HTTP server that can handle large amounts of traffic and data. It’s fast, secure, and reliable, having been around since 1996. Apache is modular and can be customized to improve its performance as needed.

**MySQL**: MySQL is a relational database management system (RDBMS) which means that it stores and presents data in tabular form, organized in rows and columns. It is open-source and easy to use. MySQL is scalable and can handle tremendous quantities of data, but its performance and efficiency can suffer when dealing with exceptionally large databases.

Graphical user interface, table

Description automatically generated

**Which stack should you use?**

Now that you’ve been introduced to some of the most popular web development stacks, you’re probably wondering which one is best for you. The answer depends on what your goals are, as well as your existing programming experience

If you’re new to web development, the MEAN or MERN stack are good starting points. These stacks are relatively easy to learn, and they use a single programming language ([JavaScript](https://www.educative.io/blog/javascript-data-structures?eid=5082902844932096)) which makes things simpler.

If you have more experience and are looking for a bit more flexibility, a stack like LAMP or MEVN might be better suited to your needs. These stacks give you more freedom to choose the tools and technologies that you want to use.

If you want to use JavaScript for both the front-end and back-end of your web application, then any of the JS-based stacks (MEAN, MERN, MEVN) would be a good choice. If you want to be able to use different programming languages for the front-end and back-end, then a stack like LAMP, WAMP, or MAMP might be better.

It’s also worth noting that you don’t have to strictly adhere to any of these stacks. You can mix and match different technologies if you want to. Ultimately, what’s most important is that you choose the tools and technologies that will best help you achieve your goals

**Tips for picking and mastering a stack**

When you’re new to web development, you may be tempted to go for breadth over depth when learning about the various tools and technologies available. However, it’s usually better to focus on one stack at a time so that you can master the tools and technologies it contains.

With that in mind, here are some tips for [picking (and mastering) a tech stack](https://www.educative.io/blog/react-angular-vue-comparison?eid=5082902844932096):

* **Define your interests and career goals**. From there, you’ll pick a stack that’s most relevant to your needs.
* **Start with the fundamentals**. You don’t need to know everything about a tech stack to use it. Start by getting a strong handle on the basic features of different technologies in your stack.
* **Try to experiment**. If you’re unsure which stack is right for you, try out a few different ones and see what works best.
* **Take your time to master the technologies of your chosen stack**. The better you know the tools in your stack, the more productive you’ll be as a developer.

No matter which stack you choose, remember that the most important thing is to keep learning and expanding your skillset. If you can do that, then it won’t be long before you’re making awesome web applications.

**Wrapping up and next steps**

Don’t make the mistake of thinking that you have to stick with one stack for your entire career. The beauty of web development is that there are always new technologies to learn and new ways to solve problems. So, keep your mind open, experiment with different stacks, and always be willing to adapt as the industry evolves.

*Happy learning!*

To get started learning these concepts and more, check out Educative’s [Become a Front-End Developer](https://www.educative.io/path/become-front-end-developer?eid=5082902844932096).

**Continue learning about web development on Educative**

* [What is Django Python? Build your first program from scratch](https://www.educative.io/blog/what-is-django-python?eid=5082902844932096)
* [React component guide: Class vs functional](https://www.educative.io/blog/react-component-class-vs-functional?eid=5082902844932096)
* [Node.js vs React.js: Which to choose for your web app project](https://www.educative.io/blog/nodejs-vs-reactjs?eid=5082902844932096)