https://www.upwork.com/hiring/development/back-end-web-developer/:

Get Started

Back-End Technology: The Role of the Back-End Web Developer

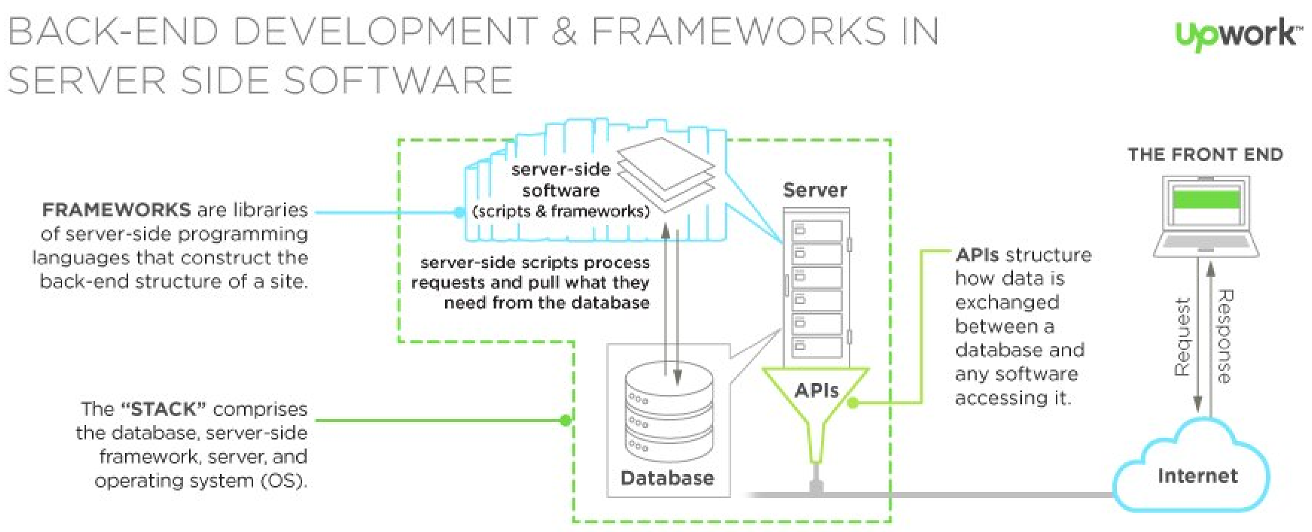
Your website or dynamic web application is a sum of layers—structure, design and content, and functionality. The technology and programming that "power" a site—what your end user doesn't see but what makes the site run—is called **the back end.** Consisting of the *server*, the *database*, and the *server-side applications*, it's the behind-the-scenes functionality—the brain of a site. This is the ecosystem of the database manager and the back-end developer.

Here's a look at the role of back-end programmers: their responsibilities, the environment they work in, the technologies they use, and related back-end skills.

Understanding the Back End: Adding Function to a Form

The back end is the machine that runs a site—the user doesn't see it or directly interact with it as with client-side technology, but it's always running in the background, delivering smooth functionality, a desktop-like experience, and information from the database right into the browser.

BACK-END DEVELOPMENT BASICS


BACK-END DEVELOPMENT & FRAMEWORKS IN SERVER SIDE SOFTWARE
upwork
FRAMEWORKS are libraries of server-side programming languages that construct the back-end structure of a site.
server-side software (scripts & frameworks)
server-side scripts process requests and pull what they need from the database
Server
THE FRONT END
APIs structure how data is exchanged between a database and any software accessing it.
Request
Response
APIs
The "STACK" comprises the database, server-side framework, server, and operating system (OS).
Database
Internet

- Back-end code adds **utility** to everything the front-end designer creates.
- The back end is a combination of a **database** and a software written in a **server-side language**, which are run on web servers, cloud-based servers, or a hybrid combination of both. A network's server set-up can vary, with the server-side workload divided up between various machines (e.g., a server dedicated to housing the database).
- This server-side application directly interacts with the database via an application programming interface (API), which pulls, saves, or changes data.
- The data are returned and converted into front-end code a user interacts with: filling out a form, creating a profile, shopping online, etc.
- In general, anything you see on a site is made possible by back-end code, which exists on, and is powered by, a server.

The Back-End Developers' Toolbox

Back-end developers create and maintain the entire back-end function outlined above. The back-end developer takes finished front-end code and gives it working functionality—for instance, making values in a drop-down menu possible by building the infrastructure that pulls values from the database.

Other responsibilities of the back end can include

- Database creation, integration, and management—e.g., MySQL, SQLite, PostgreSQL, and MongoDB. SQLite is lightweight and fast, making it a very popular alternative to a larger MySQL driver.
- Using back-end frameworks to build server-side software, like Express.js
- Web Server technologies—e.g., J2EE, Apache, Nginx (popular for static content, like images, HTML or CSS files), and IIS
- Cloud computing integration—e.g., public cloud providers like Amazon Web Services, or private cloud environments
- Server-side programming languages—like Python, Perl, PHP, Ruby, and JavaScript, when implemented with the server-side development environment, Node.js
- Operating systems: Linux- and Unix-like operating systems, MacOS X, Windows Server

- Content management system (CMS) development, deployment, and maintenance
- API integration
- Security settings and hack prevents
- Reporting—generating analytics and statistics like system reports of server load, number of visitors, geography of visitors, etc.
- Backup and restore technologies for website's files and DB.

Server-Side Programming Languages and Frameworks

Back-end developers use an array of programming languages and frameworks when building server-side software. They may choose a framework to suit their working style or a site's specific requirements. They may also work with a language within a software stack. Popular back-end technology includes

- Ruby: Great for building complicated logic on the database side of a site, Ruby bundles the back-end and database functionality that PHP and SQL can offer as a pair—it's great for startups, easy maintenance, and high-traffic demands. It requires the Ruby on Rails framework, which has vast libraries of code to streamline back-end development. *Ruby-powered sites: Hulu and Twitter*
- Java**:** A subset of the C language, Java comes with a huge ecosystem of add-on software components. At its core, Java is a variation of C++ with an easier learning curve; what's more, it's platform-independent thanks to the Java Virtual Machine. "Compile once, run anywhere" is its motto—and it's excellent for enterprise-level applications, high-traffic sites, and Android apps.
  - Frameworks include: Struts and Hibernate
- **C#:** C# is an enhanced, second-generation version of the C language, one of the earliest back-end programming languages. C# is a general-purpose, object-oriented version specifically developed by Microsoft for the .NET Framework.
- **Python:** With fewer lines of code, the Python language is fast, making it ideal for getting things to market quickly. The emphasis is on readability and simplicity, so it's great for beginners. The oldest of the scripting languages, it is powerful and works well in object-oriented designs. *Python-powered sites: YouTube, Google*
  - Frameworks include: Django, Flask, and Pyramid
- PHP**:** The most popular server-side language on the web, PHP is designed to pull and edit information in the database. It's most commonly bundled with databases written in the SQL language. PHP is unique in that it was built for the web, not adapted for it, and remains the most widely used language on the web. PHP has a number of modern frameworks as well.
- Perl**:** With 27 years of revisions and changes under its belt, Perl 5 is a high-level, general-purpose, interpreted language powerful for programming database integration with Oracle, Sybase, MySQL, and more. It runs on more than 100 platforms and is—like Python and Ruby—object-oriented and open-source.
- **Erlang:** A general-purpose programming language, Erlang is also a concurrent language, which means several processes can run simultaneously on the language-level without external library support. It's used in the LYME and LYCE stacks, numerous CMS and databases, GitHub, and Goldman Sachs's platform, supporting its high-frequency trading requirements.
- Node.js**:** This breakthrough development environment, part of the JavaScript-powered MEAN stack, allows the front-end JavaScript language to be used in server-side applications with the Express.js framework.

Back-End Software Stacks

Depending on which "stack" you choose when building the back end, your back-end developer will need cross-component expertise.

What is a software stack? “Stacks” are just bundles of software for different aspects of your site’s back end, combined for their compatibility and functionality to streamline development and deployment. The components include an **operating system,** a web server**,** a **database,** and server-side scripting language**.** You’re not limited to the components in a stack—they’re interchangeable based on your needs and customizable. *Learn more about how to choose the stack that’s right for your project.*

Two common stacks:

LAMP: Linux/Apache/MySQL/PHP

LAMP consists of free, open-source software components that work well for dynamic websites and applications. It’s the most traditional stack model, with a few variations for different operating systems, servers, and database options. In the LAMP stack, PHP is interchangeable with the languages Python and Perl.

LAMP benefits: flexible, customizable, easy to develop, easy to deploy, secure, and comes with a huge support community since it’s open source.

MEAN: MongoDB/Express.js/AngularJS/Node.js

MEAN is an all-JavaScript-powered replacement for the traditional LAMP stack. It’s excellent for businesses looking to be agile and scalable, offering flexibility with the MongoDB document-based database and lots of features for building single- and multipage web applications. By using JavaScript across the front and back ends, developers working on the client side can easily understand the server-side code, which leads to greater productivity for your team.

MEAN benefits: single language used, supports the MVC pattern, uses JSON for data transfer, offers access to Node.js’s JavaScript module library and the Express.js framework, is open source.

Learn more about the back-end developer’s role with a look at the technology of server-side scripting. Want to learn more about software stacks? Check out our Choosing the Right Software stack guide.