– [noflufffobs.com](https://nofluffjobs.com/)  
– [justjoin.it](https://justjoin.it/)  
– [pracuj.pl](https://www.pracuj.pl/)  
– [4programmers.net](https://4programmers.net/)

**Effort to learn:** The effort to learn a web framework depends on how familiar you are with the underlying programming language, the consistency of its API, the quality of its documentation, and the size and activity of its community.

**Productivity:** Productivity is a measure of how quickly you can create new features once you are familiar with the framework, and includes both the effort to write and maintain code (since you can't write new features while old ones are broken).

Many of the factors affecting productivity are similar to those for "Effort to learn" — e.g. documentation, community, programming experience, etc

* *Framework purpose/origin*: Some web frameworks were initially created to solve certain types of problems, and remain *better* at creating web apps with similar constraints. For example, Django was created to support development of a newspaper website, so it's good for blogs and other sites that involve publishing things. By contrast, Flask is a much lighter-weight framework and is great for creating web apps running on embedded devices.

Whether or not the framework encourages good development practices: For example, a framework that encourages a [Model-View-Controller](https://developer.mozilla.org/en-US/docs/Web/Apps/Fundamentals/Modern_web_app_architecture/MVC_architecture) architecture to separate code into logical functions will result in more maintainable code than one that has no expectations on developers.

**Caching support:** As your website becomes more successful then you may find that it can no longer cope with the number of requests it is receiving as users access it. At this point you may consider adding support for caching. Caching is an optimization where you store all or part of a web response so that it does not have to be recalculated on subsequent requests. Returning a cached response is much faster than calculating one in the first place.

React, Node.js, Golang, and Ruby on Rails have excellent scalability.

**Node.js**

Node.js is a popular open source JavaScript run-time environment. It was first released in 2009, and it has since then exploded in popularity

very high performance and nearly unlimited scalability.

One important downside of Node.js that all businesses should be aware of is that it lacks consistency. Its Application Programming Interface (API) changes very often, and there has been a number of backward-incompatible changes since the first version of Node.js was released in 2009

[Node.js](https://nodejs.org/en/) is JavaScript runtime built on Chrome’s V8 JavaScript engine, allowing developers to run JavaScript code on the server. What is unique to Node.js, however, is its **ability to serve data requests without waiting for the completion of another** . This asynchronous nature makes it perfect for[developing IoT web apps](https://moodup.team/blog/8-things-to-consider-before-building-an-iot-app/)that send large numbers of requests to the server at the same time

**Express** is a fast and minimalist framework built on NodeJS. It enables the easy creation of REST APIs by allowing you to define the code that runs for various requests in the server.

Express is extremely popular, partially because it eases the migration of client-side JavaScript web programmers into server-side development

Because Express is a minimalist web framework it does not incorporate every component that you might want to use (for example, database access and support for users and sessions are provided through independent libraries). There are many excellent independent components, but sometimes it can be hard to work out which is the best for a particular purpose!

NodeJS is the most convenient platform for hosting a React application for the following two reasons:

1. With the use of NPM (Node Package Manager), NodeJS works in parallel to the NPM registry for the convenient installation of any package through NPM CLI.
2. Node packages the React application as a single file for easy execution using webpack and some other Node modules.

## Ruby on Rails

Ruby on Rails is a server-side web application framework written in Ruby.

Ruby on Rails, however, will be a good choice as it provides DSL (Digital Subscriber Line), helping you configure a content security policy for your app

high speed of development, vibrant community, fantastic tooling, and strong adherence to standards.

Ruby on Rails framework, which provides access to a set of basic libraries, will save significant time.

slow runtime speed, lack of flexibility, and the high cost of wrong decisions have made it less desirable in the eyes of many businesses.

## [Django](https://www.djangoproject.com/)

Django is a high-level Python framework maintained by the Django Software Foundation (DSF).

Django (it is one of the easiest to learn based on the above criteria).

Django follows the "Batteries included" philosophy and provides almost everything most developers might want to do "out of the box"

[Django](https://www.djangoproject.com/) is a high-level python web framework that is built with Speed, Security, and Scalability at its core.

It also encourages RAD (Rapid Application Development ) using clean, pragmatic software design.

## Security:

Security is primary in Django as it helps developers to avoid some securities pitfalls.

## Scalability:

Scalability is primary in the framework making it easy for web applications to scale efficiently.

[Django](https://www.djangoproject.com/) is a high-level Python Web framework **designed for rapid development and scaling** . It’s **very versatile** and can be used to developing solutions for content management and scientific computing platforms. Django is also very **easy to learn** , **understand** , use and **places an emphasis on speedy product development**, making it ideal for small to medium projects.

Django, however, is rather monolithic and **does not allow for the division of a project into smaller parts** as with Node.js. This might prove to be a disadvantage when working on large projects that need to be divided into several parts between developers. It’s also very strongly opinionated and **does not allow developers much freedom** as with Node.js.

### Spring

[Spring](https://spring.io/) is another successful solution build on Java, the most popular programming language in the world due to its object-oriented robustness. This popular framework for creating enterprise applications **comes packed with many features and additions** that allow a developer to start an application with a basic website and authorization, almost out of the box. Spring’s **modularity gives developers the freedom** to pick and ignore classes, helping them focus more on customer-centric development and less on reinvention.

Spring, however, is very configuration dependent and **requires a developer to write a sizable amount of boilerplate code in order to add new features** . It also requires developers to undergo a **steep learning curve** as there isn’t much set standards and best working practices. This is one possible reason why you might encounter Spring when working with more experienced, bigger teams working on enterprise projects.

Laravel:

## Eloquent Database ORM:

This is one of the most powerful features of Laravel, Eloquent takes away the pain of manually handling database away and provides an easy to use ORM. Check out the [complete guide on Laravel relationships](https://masteringbackend.com/posts/complete-guide-on-laravel-relationships/) to give you a clear picture of how Laravel handles database operations easily.