<https://hackernoon.com/top-10-python-web-frameworks-to-learn-in-2018-b2ebab969d1a>

Frameworks make developers’ lives easier by offering them a structure for application development. They automate the implementation of common solutions, cutting development time and allowing developers to focus more on application logic instead of routine elements.

In this article, we share our own list of the top ten Python web frameworks that we believe will be useful on your way to becoming a professional backend developer and improving your existing skill set. Keep in mind that these frameworks are listed in no particular order: we’ve focused on showing what you’re able to do as opposed to telling what you should do.

Some things to consider

When deciding which framework to use, look at the size and complexity of your project. If what you’re looking to develop is a large system packed with features and requirements, a full-stack framework might be the right choice. If your app is on the smaller and simpler side, you should probably consider a microframework.

You can find information about the type and focus of some frameworks [here](https://wiki.python.org/moin/WebFrameworks). A final decision, though, should come from your own understanding of your project and the tasks you want to simplify.

However, frameworks can also stand in the way of development. When choosing a full-stack framework, you’re often signing up for a set of limitations. Of course, you can find ways to work around them, but be careful you don’t spend more time fighting for your own freedom than you would have writing an app in pure Python.

Full-stack frameworks

Django

[Official Website](https://www.djangoproject.com/) | [GitHub](https://github.com/django) | [PyPI](https://pypi.python.org/pypi/Django/1.11.1) | [Awesome](https://github.com/rosarior/awesome-django)

GitHub Stars: 31,592 | GitHub Forks: 13,361

Django is a free open-source full-stack Python framework. It tries to include all of the necessary features by default as opposed to offering them as separate libraries.

Some of the exemplary features of Django are its [authentication](https://docs.djangoproject.com/en/dev/topics/auth/), [URL routing,](https://docs.djangoproject.com/en/dev/topics/http/urls/)[template engine](https://www.fullstackpython.com/django-templates.html), [object-relational mapper](https://www.fullstackpython.com/object-relational-mappers-orms.html) (ORM), and [database schema migrations](https://docs.djangoproject.com/en/dev/topics/migrations/) (Django v.1.7+).

Django uses its ORM to map objects to database tables. The same code works with different databases and isn’t difficult to transfer from one database to another. The main databases that Django works with are [PostgreSQL](https://www.postgresql.org/), [MySQL](https://www.mysql.com/), [SQLite](https://www.sqlite.org//index.html), and [Oracle](https://www.oracle.com/index.html), but third-party drivers allow you to use others as well.

Here at SteelKiwi, we use Django as our primary framework for backend development. If you’d like to see the amazing apps we’ve already developed with Django, visit our [Projects page.](https://steelkiwi.com/projects/)

Pyramid

[Official Website](https://trypyramid.com/) | [GitHub](https://github.com/Pylons/pyramid) | [PyPI](https://pypi.python.org/pypi/pyramid) | [Awesome](https://github.com/uralbash/awesome-pyramid)

GitHub Stars: 2,640 | GitHub Forks: 782

Pyramid is an open-source Python-based web application framework. Its main goal is to do as much as possible with minimum complexity.

The most striking feature of Pyramid is its ability to work well with both small and large applications. Some of Pyramid’s great features include:

Single-file applications

[URL generation](https://docs.pylonsproject.org/projects/pyramid/en/latest/api/url.html)

[Extensible configuration](https://trypyramid.com/resources-extending-pyramid.html)

All-embracing [templating](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/templates.html) and [asset specifications](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/assets.html)

Flexible [authentication](https://docs.pylonsproject.org/projects/pyramid/en/latest/api/authentication.html) and [authorization](https://docs.pylonsproject.org/projects/pyramid/en/latest/api/authorization.html)

[Testing](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/testing.html), support, and [comprehensive data documentation](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/introduction.html)

[View predicates](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/viewconfig.html) and many per-route views

[Function decorators](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/viewconfig.html#mapping-views-using-a-decorator-section)

[Predicates](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/viewconfig.html#view-configuration-parameters)

[Renderers](https://docs.pylonsproject.org/projects/pyramid/en/latest/narr/renderers.html)

TurboGears

[Official Website](http://www.turbogears.org/) | [GitHub](https://github.com/TurboGears/tg2/) | [PyPI](https://pypi.python.org/pypi/TurboGears2/2.3.11)

GitHub Stars: 230 | GitHub Forks: 56

TurboGears is an open-source, data-driven full-stack web application framework.

It allows you to quickly develop extensible data-driven web applications. TurboGears comes with user-friendly templating and a powerful and flexible ORM.

Some of TurboGears’ distinctive features include:

Multi-database support

MVC-style architecture

Support for [SQLObject](https://pypi.python.org/pypi/SQLObject) and [SQLAlchemy](https://www.sqlalchemy.org/)

[Kid](http://www.kid-templating.org/) and [Genshi](https://genshi.edgewall.org/) included in preferred templating languages

Validation with [FormEncode](http://www.formencode.org/en/latest/)

[Pylons](https://pylonsproject.org/) as a web server

[ToscaWidgets,](http://www.toscawidgets.org/) an application library that simplifies coordination of frontend design and server development

[PasteScript](https://pypi.python.org/pypi/PasteScript) templates

Front-facing WSGI-based servers ([Paste HTTP server](http://paste.readthedocs.io/en/latest/paste-httpserver-threadpool.html), [CherryPy WSGI/HTTP server,](http://cherrypy.org/) etc.)

Command-line tools

[MochiKit JavaScript library](https://mochi.github.io/mochikit/) integration

All features implemented as function decorators

Web2py

[Official Website](http://www.web2py.com/) | [GitHub](https://github.com/web2py) | [PyPI](https://pypi.python.org/pypi/web2py)

GitHub Stars: 1,536 | GitHub Forks: 742

Web2py is a scalable open-source full-stack Python framework. However, you should know before reading further that Web2py does not support Python 3.

What’s great about Web2py, though, is that it comes with its own web-based IDEwhich, among other things, includes a code editor, debugger, and one-click deployment.

Other valuable Web2py features include:

No requirements for installation and configuration

Ability to run on Windows, Mac, Linux/Unix, [Google App Engine](https://cloud.google.com/appengine/docs/), [Amazon EC2](https://aws.amazon.com/ec2/), and any web hosting that supports either Python 2.5–2.7 or Java+Python

Readability of multiple protocols

Data security that prevents such vulnerabilities as cross-site scripting, injection flaws, and malicious file execution

Successful employment of software engineering practices, making code easy to read and maintain

Error tracking, thorough error logging and ticketing

[Role-based access control](http://web2py.com/books/default/chapter/29/09/access-control)

Support for [internationalization](http://web2py.com/books/default/chapter/29/04/the-core#Internationalization--and-Pluralization-with-T)

Backward compatibility that ensures user-oriented advancement without the need to lose ties with earlier versions

Microframeworks

Flask

[Official Website](http://flask.pocoo.org/) | [GitHub](https://github.com/pallets/flask/) | [PyPI](https://pypi.python.org/pypi/Flask/0.12) | [Awesome](https://github.com/humiaozuzu/awesome-flask)

Github Stars: 32,959 | GitHub Forks: 10,254

Flask is a Python framework available under the BSD license. It was inspired by the Sinatra Ruby framework. Flask depends on the [Werkzeug WSGI toolkit](http://werkzeug.pocoo.org/) and [Jinja2](http://quintagroup.com/cms/python/jinja2)template.

The main idea behind Flask is to help build a solid web application foundation. From there, you can use any extensions you might need.

Flask’s lightweight and modular design makes it easily adaptable to developers’ needs. It includes a number of useful out-of-the-box features:

Built-in [development server](http://flask.pocoo.org/docs/0.12/server/) and a fast debugger

Integrated support for [unit testing](http://flask.pocoo.org/docs/0.12/testing/)

[RESTful request dispatching](http://flask-restful.readthedocs.io/en/0.3.5/quickstart.html)

[Jinja2 templating](http://flask.pocoo.org/docs/0.12/templating/)

Secure cookies support (client-side sessions)

WSGI 1.0 compliance

Unicode-based

Ability to plug in any ORM

HTTP request handling

Bottle

[Official Website](http://bottlepy.org/docs/dev/index.html) | [GitHub](https://github.com/bottlepy/bottle) | [PyPI](https://pypi.python.org/pypi/bottle/0.12.13)

GitHub Stars: 5,198 | GitHub Forks: 1,065

Bottle is a microframework. Originally meant for building APIs, Bottle implements everything in a [single source file](https://github.com/bottlepy/bottle/blob/master/bottle.py). It has no dependencies apart from the [Python Standard Library.](https://docs.python.org/3/library/) Coding with Bottle gets you closer to the metal than coding with any full-stack framework.

Its default features include routing, templating, utilities, and a basic abstraction over the [WSGI](http://wsgi.readthedocs.io/en/latest/) standard.

Routing — supports requests to function-call mapping, allowing you to achieve clean and dynamic URLs

Templates — a fast and Pythonic [template engine](http://bottlepy.org/docs/dev/tutorial.html#tutorial-templates) out of the box along with full support for [mako](http://www.makotemplates.org/), [jinja2](http://jinja.pocoo.org/), and [cheetah](http://cheetahtemplate.org/)

Utilities — comfortable access to form data, file uploads, cookies, headers, and other HTTP-related metadata

Server — a built-in HTTP development server that supports [fapws3](https://github.com/william-os4y/fapws3), [bjoern](https://github.com/jonashaag/bjoern), [GAE](https://cloud.google.com/appengine/docs/), [СherryPy](http://cherrypy.org/), and any other WSGI-capable HTTP server

Bottle is a perfect solution for prototyping, learning the organization of web frameworks, and building simple personal apps.

CherryPy

[Official Website](http://cherrypy.org/) | [GitHub](https://github.com/cherrypy/cherrypy) | [PyPI](https://pypi.python.org/pypi/CherryPy)

GitHub Stars: 595 | GitHub Forks: 161

CherryPy is an open-source, minimalist web framework. It makes building Python web applications no different than building any other object-oriented program.

In fact, a CherryPy-powered web app is a standalone Python application that embeds its own multi-threaded web server. CherryPy apps run on any operating system that supports Python (Windows, macOS, Linux, etc.). They can be deployed anywhere you can run an ordinary Python app. You don’t need [Apache](https://www.apache.org/) for CherryPy apps, but you can run a CherryPy app behind Apache just as well as you can run it behind [Lighttpd](https://www.lighttpd.net/) or [IIS](https://www.iis.net/).

CherryPy isn’t a very limiting framework, as it allows you to use any type of technology for templating, data access, and so on. However, it’s still able to handlesessions, statics, cookies, file uploads, and everything else that a web framework typically can.

Some of the default CherryPy features include:

An [HTTP/1.1-compliant WSGI](https://www.w3.org/Protocols/rfc2616/rfc2616-sec1.html#sec1.2) thread-pooled webserver

Simplicity of [running multiple HTTP servers](http://docs.cherrypy.org/en/latest/advanced.html#multiple-http-servers-support) at once

A [powerful configuration system](http://docs.cherrypy.org/en/latest/basics.html#config)

A [flexible plugin system](http://docs.cherrypy.org/en/latest/extend.html#server-wide-functions)

Out-of-the-box tools for [caching](http://docs.cherrypy.org/en/latest/pkg/cherrypy.lib.html#module-cherrypy.lib.caching), [encoding](http://docs.cherrypy.org/en/latest/_modules/cherrypy/lib/encoding.html), [sessions](http://docs.cherrypy.org/en/latest/basics.html#using-sessions), [authentication](http://docs.cherrypy.org/en/latest/basics.html#authentication), [static content](http://docs.cherrypy.org/en/latest/basics.html#static-content-serving), and [so on](http://docs.cherrypy.org/en/latest/extend.html#per-request-functions)

Built-in support for [profiling](http://docs.cherrypy.org/en/latest/pkg/cherrypy.lib.html?highlight=profiler#module-cherrypy.lib.profiler), [coverage](http://docs.cherrypy.org/en/latest/pkg/cherrypy.lib.html?highlight=cover#module-cherrypy.lib.covercp), and [testing](http://docs.cherrypy.org/en/latest/advanced.html#testing-your-application)

Ability to run on [Python 2.7+](https://www.python.org/download/releases/2.7/), [Python 3.1+](https://www.python.org/download/releases/3.1/), [PyPy](https://pypy.org/), [Jython](http://www.jython.org/), and [Android](http://www.defuze.org/archives/228-running-cherrypy-on-android-with-sl4a.html)

Asynchronous frameworks

Sanic

[Official Website](http://sanic.readthedocs.io/en/latest/) | [GitHub](https://github.com/channelcat/sanic) | [PyPI](https://pypi.python.org/pypi/Sanic)

GitHub Stars: 8,488 | GitHub Forks: 776

Sanic is a Python web framework built on [uvloop](https://github.com/MagicStack/uvloop) and created specifically for fast HTTP responses via asynchronous request handling.

It runs on [Python 3.5+.](https://www.python.org/downloads/release/python-350/) Sanic supports asynchronous request handlers, which makes it compatible with Python 3.5’s async/await functions. This enhances its speed, offering non-blocking capabilities.

In a benchmark test with one process and 100 connections, Sanic was able to handle [33,342 requests](https://github.com/channelcat/sanic) per second.

Tornado

[Official Website](http://www.tornadoweb.org/en/latest/) | [GitHub](https://github.com/tornadoweb/tornado) | [PyPI](https://pypi.python.org/pypi/tornado)

GitHub Stars: 15,126 | GitHub Forks: 4,432

Tornado is a Python web framework and asynchronous networking library. It uses a non-blocking network I/O and solves the C10k issue (meaning that, if configured properly, it can handle 10,000+ concurrent connections).

This makes it a great tool for building applications requiring high performance and tens of thousands of concurrent users.

Tornado’s main features are:

Built-in support for [user authentication](http://www.tornadoweb.org/en/stable/guide/security.html)

Real-time services

High-quality performance

Python-based [web templating](http://www.tornadoweb.org/en/stable/guide/templates.html) language

Non-blocking [HTTP client](http://www.tornadoweb.org/en/stable/httpclient.html)

Implementation of [third-party authentication and authorization](http://www.tornadoweb.org/en/stable/auth.html) schemes ([Google OpenID/OAuth](https://developers.google.com/identity/protocols/OpenIDConnect), Facebook Login, [Yahoo BBAuth](https://developer.yahoo.com/auth/), FriendFeed OpenID/OAuth, [Twitter OAuth](https://developer.twitter.com/en/docs/basics/authentication/overview/oauth))

Support for [translation and localization](http://www.tornadoweb.org/en/stable/locale.html)

Other

Dash

[Official Website](https://plot.ly/products/dash/) | [GitHub](https://github.com/plotly/dash/) | [PyPI](https://pypi.python.org/pypi/dash) | [Awesome](https://github.com/Acrotrend/awesome-dash)

GitHub Stars: 3,903 | GitHub Forks: 360

Dash is an open-source Python framework used for building analytical web applications. It’s especially good for Python data scientists who aren’t very familiar with web development.

Dash applications are web servers that run [Flask](http://flask.pocoo.org/) and communicate with JSON packets over HTTP requests. Their frontend renders components with [React.js.](https://reactjs.org/)

Apps developed with Dash are rendered in the web browser and can be deployed to servers. This also means that Dash applications are inherently cross-platform and mobile-ready.

Dash developers have access to the underlying Flask instance and all of its configurable properties. To extend the capabilities of Dash applications, developers can also use the rich set of [Flask Plugins.](http://flask.pocoo.org/extensions/)

Choosing a Python web framework

The right framework might be all you need to develop a successful project. We really hope that you’ve found an interesting framework in this article. [We also have another one, the 17 best Python web frameworks of 2017,](https://steelkiwi.com/blog/best-python-web-frameworks-to-learn/) if you’re interested in exploring more options.

We’d love if you visited our [corporate GitHub page](https://github.com/steelkiwi). Our [SteelKiwi](https://steelkiwi.com) developers are trying hard to innovate in the world of tech, and the more people who get involved, the better.