Comparison of Web Frameworks on Python and Node.js Platforms

Ekaterina Eliseeva[0000−0002−4616−6829], Ilkhom Khafizov[0000−0002−9340−5239], and Ilya Gosudarev[0000−0003−4236−5991]

ITMO University, Saint Petersburg, Russia

Abstract. An important task when developing a server web application is to choose the right technological stack. The existence of a large number of Node.js and Python frameworks causes the problem of making the choice between them. This article compares the most popular frameworks by such criteria as code quality, support, performance. The first section explains the selection of frameworks for the study. The second section contains the results of the frameworks source code analysis. The third section compares framework performance based on test results. In conclusion, recommendations will be given on the choice of a framework based on the results of the comparison.

Keywords: web frameworks · Node.js · Python · web server · framework performance

1 Introduction

Nowadays, JavaScript and Python are the most popular programming languages. Python is a high-level general-purpose programming language that can be used in the development of all types of software, from simple automation scripting to system programming, game development and web applications [1]. JavaScript is the only language that works on the client site. Node.js is an open-source JavaScript execution environment that allows to run JavaScript on the server side as well.

Web-based server frameworks make it easy to write, maintain and scale web applications. They provide tools and libraries that simplify common web development tasks, such as URL routing, database interaction, session support, and user authentication.

The performance of a web application directly depends on the right choice of technological stack, so the choice of language and infrastructure for the development of the server part of the application must be careful and thoughtful. The existence of a large number of Node.js and Python frameworks causes the problem of making the choice between them, especially for beginners. Therefore, it is necessary to conduct a comparative analysis of frameworks, to highlight important advantages and disadvantages.

2 The choice of frameworks

To compare Python and Node.js server frameworks, the frequency of frameworks usage on these platforms was analyzed (Fig. 1 and Fig. 2) .

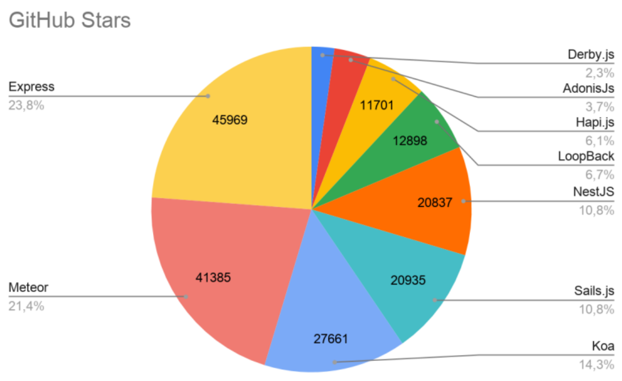

Fig. 1. GitHub Stars of Node.js frameworks.

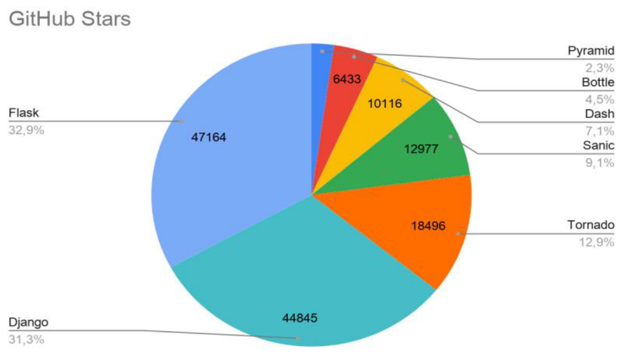

Fig. 2. GitHub Stars of Python frameworks.

The most frequently used Express and Flask micro frameworks and Sails.js and Django frameworks using the MVC pattern were selected. Table 1 shows GitHub Stars and the number of projects on GitHub that use Express, Sails.js, Flask and Django frameworks.

Table 1. Framework Rating on GitHub.

	Express	Sails.js	Flask	Django
GitHub Stars	45 919	20 922	47 164	44 845
Used by	5 115 779	20 127	333 799	312 868

From the diagrams and tables we can see that Express is the most popular among Node.js frameworks and the absolute leader in the number of uses in other repositories [2], Sails.js is one of the most frequently used Node.js MVC frameworks [3], Flask and Django are in the same degree ahead of other Python frameworks in terms of number of stars and use in other projects [4].

3 Comparative analysis

3.1 Framework code analysis

The following programs were used to analyze framework code:

– SLOC https://github.com/flosse/sloc - to get the number of lines and comments in the framework code;
– complexity-report https://github.com/escomplex/complexity-report - to calculate Halstead's metrics, cyclomatic complexity, support index;
– Radon https://pypi.org/project/radon - to calculate the framework source code metrics on Python.

Table 2 shows the results of calculating the framework source code metrics.

Table 2. Framework source code analysis.

Metric	Express	Sails.js	Flask	Django
Number of lines of code	13 013	203 936	10 486	271 045
Number of comments	5 635	106 244	4 255	58 742
Percentage of comments	43.3%	52.1%	40.6%	21.7%
Mean per-function Halstead effort	2 345	2 475	510	2 324
Mean per-function cyclomatic complexity	3.244	2.230	3.058	1.614
Maintainability index	67	82	68.913	87.64

The highest percentage of code documentation has Sails.js; Express and Flask in this category also have good indicators. Express and Flask have high cyclomatic complexity, which makes their support more complex. Django doesn't have a lot of comments, but as well as Sails.js it has the lowest cyclomatic complexity and high support index, which means easier code support.

Sails.js, Express and Flask have better documentation than Django, but their cyclomatic complexity is higher and the support index is lower, making their overall performance similar.

3.2 Performance analysis

Framework performance tests were conducted on applications with the same functionality, with the following versions of the environment and frameworks:

- Node v8.10.0, Express 4.17.1, Sails 1.2.3;
- Python 3.6.8, Flask 1.1.1, Django 2.2.6;

The results of the performance study are also affected by the software and hardware used, which are as follows in this analysis:

- Hardware: dual-core CPU Intel Core i5-3337U @ 1.80GHz, 6GB RAM;
- Software - Ubuntu operating system 18.04.3 LTS 64-bit;

For the first test the application which returns a sequence of Fibonacci numbers for the parameter passed in URL has been created. The main performance indicators were identified as Requests per second, Time per request. These indicators were calculated using the Apache Benchmark web server testing tool https://httpd.apache.org/docs/2.4/programs/ab.html [5].

Table 3 shows the minimum processing time of a request, the time intervals for which 50%, 75% and 95% of the requests were fulfilled, the number of requests per second and the average processing time of one request. A graph showing the number of completed requests to the server that returns a sequence of Fibonacci numbers for a certain time is shown in Figure 3.

Table 3. Runtime and the number of requests to the server with Fibonacci.

	min, ms	50%, ms	75%, ms	95%, ms	Requests per second (mean)	Time per request (mean), ms
Express	3	4	5	7	2019.35	4.952
Sails.js	11	14	16	23	638.40	15.664
Flask	8	18	21	26	524.58	19.063
Django	13	38	43	50	263.66	37.928

The Flask and Django frameworks have different implementations of working with JSON, therefore, the second test compares the performance of frameworks when working with JSON. The source data is an object that is converted to the 'application / json' format and returned as response. Table 4 and the graph in Figure 4 show the execution time of requests and the number of completed requests to the server returning JSON.

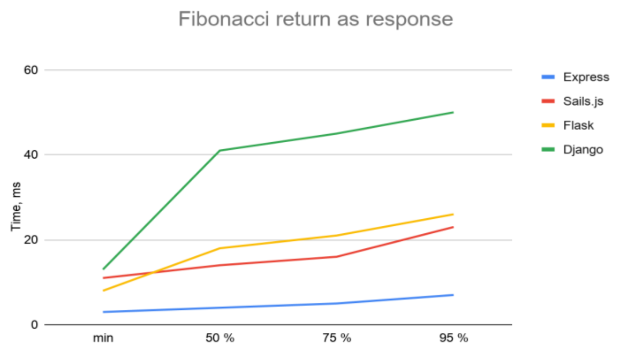

Fig. 3. Fibonacci return as response.

Table 4. Runtime and the number of requests to the server sending JSON.

	min, ms	50%, ms	75%, ms	95%, ms	Requests per second (mean)	Time per request (mean), ms
Express	4	5	6	9	1757.67	5.689
Sails.js	10	14	16	21	642.28	15.569
Flask	6	22	24	267	452.57	22.096
Django	5	34	37	43	293.35	34.089

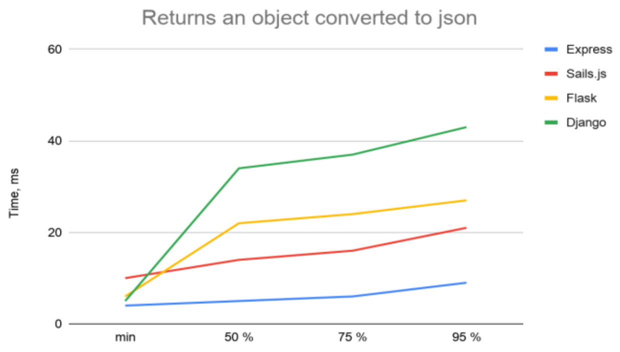

Fig. 4. JSON returns as response.

Express has the best performance indicators, the average number of processed requests which is more than twice as many as Sails.js. Flask also has good results, but is a little behind Sails.js. Django showed the slowest results according to the results of the analysis.

4 Results

The comparative analysis presented in this paper allows to make certain recommendations for choosing the right framework. According to the results of the frameworks source code analysis, Sails.js and Django have the best indicators. Performance tests have shown that the fastest framework is Express, in second place is Sails.js. Flask and Django frameworks are slower, but the average number of requests per second from Flask is almost twice as high as Django.

So the problem of choice that has been put forward in the introduction can be solved according to the recommendations that follow. If a developer has programming experience in JavaScript, it is logical to choose one of the Node.js frameworks and avoid the time-consuming learning. Alternatively, the choice of Python may be more optimal because it is considered the most comprehensible and quickly learned language. For small projects it is better to choose Express or Flask, they allow to create a simple application and expand it if necessary. For larger projects the Sails.js or Django should be used, which allow to create a flexible, secure, scalable application by using built-in tools.

References

1. Volobuev N.G. Sravnenie Web freimvorkov Python. Al'manah nauchnyh rabot molodyh uchenyh XLVI uchebnoj i nauchno-metodicheskoj konferencii Universiteta ITMO. Tom 4, 44-46.
2. Express vs Flask, https://www.slant.co/versus/1277/1398/~express-js_vs_flask. Last accessed 20 Oct 2019.
3. Sails.js vs Django, https://www.slant.co/versus/1283/1746/~sails-js_vs_django. Last accessed 20 Oct 2019.
4. Comparison Of Four Popular Node.js Frameworks, https://www.toptal.com/nodejs/nodejs-frameworks-comparison. Last accessed 22 Oct 2019.
5. Testirovanie proizvoditel'nosti web serverov. Apache Benchmark, https://admins.su/site-speed-ab. Last accessed 22 Oct 2019.