ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ / INFORMATICS AND INFORMATION PROCESSES

DOI: https://doi.org/10.23670/IRJ.2023.137.29

ИССЛЕДОВАНИЕ ОПТИМИЗАЦИИ ПРОИЗВОДИТЕЛЬНОСТИ В PHP: СРАВНИТЕЛЬНЫЙ АНАЛИЗ БЭКЕНД-ФРЕЙМВОРКОВ

Научная статья

Кочнев А.А.[1, *]
[1] ORCID : 0009-0004-0106-1208;
[1] Your Next Agency, Сан-Диего, Соединенные Штаты Америки

* Корреспондирующий автор (drdispool[at]gmail.com)

Аннотация

Статья посвящена проведению сравнительного анализа PHP-фреймворков в контексте оптимизации производительности языка PHP. В качестве фреймворков, представленных к сравнению, были выбраны Laravel, Symfony, CodeIgniter, Yii и Zend – пять самых популярных бэкенд-фреймворков. Были использованы три различных системы для тестирования: низконагруженная система (Intel i5, 8GB RAM), средненагруженная система (Intel i7, 16GB RAM), и высоконагруженная система (2x Intel Xeon, 64GB RAM). База данных на каждой системе была стандартной MySQL версии 8.0, и были включены веб-серверы Apache и Nginx. В ходе исследования было проведено тестирование производительности фреймворков, включающее в себя такие сравнительные метрики, как: время ответа на запрос страницы, запись и чтение из базы данных, обработка параллельных запросов, а также анализ использования памяти и др. В работе уточняется, что одним из немногих способов сравнения языков программирования или их конфигураций остается создание единых условий и их типовое описание, как определенных задач, с которыми сталкивается программист при задачах разработки. Как итог, сравнение приобретает более объективный характер и позволяет говорить не о преимуществе языка и конкретной конфигурации в целом, а их вероятной большей эффективности в рамках описанного спектра задач. Так, исследование было направлено на выявление лучшего фреймворка с точки зрения производительности и предоставление четких данных для выбора оптимального решения при разработке веб-приложений на PHP. По итогам исследования делается вывод о том, что Laravel является наиболее производительным PHP-фреймворком, с учетом всех рассмотренных метрик. Однако это не означает, что другие фреймворки не могут быть подходящими для определенных задач и функций.

Ключевые слова: PHP, бэкенд-фреймворки, производительность, тестирование производительности, время ответа на запрос, запись в базу данных, чтение из базы данных, параллельные запросы, использование памяти.

PERFORMANCE OPTIMIZATION STUDY IN PHP: A COMPARATIVE ANALYSIS OF BACKEND FRAMEWORKS

Research article

Kochnev A.A.[1, *]
[1] ORCID : 0009-0004-0106-1208;
[1] Your Next Agency, San Diego, USA

* Corresponding author (drdispool[at]gmail.com)

Abstract

This article is dedicated to a comparative analysis of PHP frameworks in the context of optimizing PHP language performance. Laravel, Symfony, CodeIgniter, Yii and Zend – the five most popular backend frameworks – were chosen as the frameworks for comparison. Three different systems were used for testing: a low-load system (Intel i5, 8GB RAM), a medium-load system (Intel i7, 16GB RAM), and a high-load system (2x Intel Xeon, 64GB RAM). The database on each system was standard MySQL version 8.0, and Apache and Nginx web servers were included. The study involved performance testing of the frameworks, including comparative metrics such as: page request response time, record and reading from the database, parallel query processing, and memory usage analysis, among others. The work specifies that one of the few ways to compare programming languages or their configurations remains the creation of common conditions and their typical description as certain tasks that a programmer faces in development tasks. As a result, the comparison becomes more objective and allows to speak not about the advantage of a language and a particular configuration in general, but about their probable greater efficiency within the described range of tasks. Thus, the study was aimed at identifying the best framework in terms of performance and providing clear data for choosing the best solution when developing PHP web applications. The study concludes that Laravel is the best performing PHP framework, based on all the metrics considered. However, this does not mean that other frameworks may not be suitable for certain tasks and functions.

Keywords: PHP, backend frameworks, performance, performance testing, query response time, database record, reading from database, parallel queries, memory usage.

Введение

Сегодня язык программирования PHP остается одним из самых популярных языков программирования для разработки веб-приложений, что является следствием гибкости и наличия мощных бэкенд-фреймворков. Прошлые исследования автора настоящей статьи указывают на существование обширного числа бэкенд-фреймворков для актуального языка PHP, каждый из которых универсален для собственных задач и функций, обладает характерными

достоинствами и недостатками [4]. Тем не менее актуальным вопросом, несмотря на широту применения PHP, остается вопрос производительности и эффективности, который приобретает сравнительный характер при рассмотрении других языков программирования или фреймворков в рамках данного языка. В случае, если сравнительный анализ производительности языков программирования имеет неоднозначный характер, т.к. каждый язык, бесспорно, обладает собственными достоинствами и недостатками, ориентирован на конкретные задачи, а оценка приобретает больше субъективных сторон, то сравнение бекенд-фреймворков на базе конкретного языка программирования приобретает более объективную характеристику, поскольку позволяет выбрать исполнение для целей разработки, а также учесть более стандартизированные метрики [9]. На примере разработки веб-приложений, выбор бэкенд-фреймворка оказывает определенное влияние на множество факторов, включая производительность, стабильность, безопасность, и возможности масштабирования приложения. Учитывая существование огромного разнообразия фреймворков, доступных для PHP, из числа которых наиболее популярными становятся Laravel, Symfony, CodeIgniter, Zend, Yii и др., необходимость приобретает проведение сравнительного анализа для целей выбора того фреймворка, который положительно сказывается на производительности языка PHP.

Цель исследования – провести сравнительный анализ PHP-фреймворков в контексте оптимизации производительности языка PHP.

Задачи исследования:

1) охарактеризовать теоретические основы повышения производительности языков программирования;

2) провести тестирование языка PHP на базе различных фреймворков с учетом выделенных метрик;

3) сгруппировать полученные данные и сделать выводы о сильных и слабых сторонах каждого фреймворка.

Методология исследования

Теоретическим базисом настоящего исследования послужили работы отечественных авторов, посвященные процессам оптимизации языков программирования для решения конкретных типов задач. В работе мы используем методы как теоретического исследования: анализ, синтез, сравнение и др., так и реализуем специальные методы компьютерного тестирования. Так, нами были использованы три различных системы для тестирования: низконагруженная система (Intel i5, 8GB RAM), средненагруженная система (Intel i7, 16GB RAM), и высоконагруженная система (2x Intel Xeon, 64GB RAM). База данных на каждой системе была стандартной MySQL версии 8.0, и были включены веб-серверы Apache и Nginx. Для целей измерения производительности были выбраны следующие метрики: время ответа на запрос страницы, время записи в базу данных (БД), время чтения из БД, и производительность при параллельных запросах. Кроме этого, мы исследовали использование памяти, проводили нагрузочное тестирование, и описывали другие релевантные метрики, чтобы получить полное представление о сильных и слабых сторонах каждого фреймворка.

Литературный обзор

Тема выбора языка программирования и сравнения различных конфигураций для языков остается актуальной ввиду формирования обширной системы доступных языков, каждый из которых обладает характерными достоинствами или недостатками, рекомендуется для конкретных ситуаций применения. Молодой разработчик нередко сталкивается с ситуацией, когда встает на перепутье выбора конкретного языка из множества существующих. Более того, даже в случае выбора языка, данная проблема сохраняет свое значение, поскольку встает задача выбора наиболее эффективного фреймворка. Как показывает широкая практика автора настоящего исследования в области программирования, выбор конкретных решений носит субъективный характер и часто строится на ощущениях конкретного специалиста. Как итог, отсутствуют достоверные данные, опираясь на которые можно было бы выбирать различные конфигурации фреймворков для решения тех или иных задач в области программирования [10].

Сравнение языков программирования детально представлено в трудах многих авторов. В каждом из случаев авторы исследования фокусируются на конкретной задаче, функциях и требуемых к исполнению операциях, что позволяет создать объективные условия для сравнительного анализа.

Например, в исследовании М.Ю. Егорова, С.М. Егорова, Д.М. Егорова раскрываются условия, при которых изменяется не только язык программирования, но и конфигурация самой системы. Авторы, исследуя решение задачи численного моделирования переходных внутрикамерных процессов на ракетном двигателе, фокусируются на сравнении вычислительной производительности с учетом изменения конфигурации системы. Результаты проведенного исследования авторов демонстрируют, что использование программного комплекса, который основан на другом языке программирования, способно значительно повлиять на производительность вычислений, а именно увеличить её более чем в десять раз [3]. Столь значительные результаты, зачастую, могут определенно положительно сказаться на работе программы, что требует сравнения вариантов исполнения программного кода.

Решение схожей прикладной задачи раскрыто в исследовании Е.К. Гребенниковой и П.И. Кандалова, которые фокусируются на ситуации оптимизации времени работы программного алгоритма за счет выбора одного из языков программирования. Авторы осуществляют разработку двух вариаций программного обеспечения на базе разных языков, определяя, что функциональная сторона программного кода позволяет увеличить скорость работы системы в шесть раз. Таким образом, в единой задаче при общих мощностях один язык программирования может показывать большую скорость, нежели другой [1].

Однако важно понимать, что это не отражает преимущество одного языка над другим, поскольку производительность зависит от целой системы факторов. На данный факт также указывают в своем исследовании В.В. Рокотянская и В.С. Абрамов, которые проводят схожие сравнения языков программирования и их конфигураций в рамках различных задач. Авторы приходят к выводу о том, что на основе подобных сравнений нельзя делать вывод о превосходстве одного языка над другим, поскольку необходимо сравнивать не общий итог, а каждый отдельный случай в работе языка. И хотя определенный язык может демонстрировать лучшее решение задач в большей части случаев, в

ряде других ситуаций он также может демонстрировать гораздо меньшую эффективность. Подобные противоречия не позволяют дать полностью объективную оценку производительности языков, а учет всех влияющих факторов остается попросту невозможным, ввиду их широты [6]. Например, проведенное сравнение языков программирования в работе А.В. Давыдова, А.К. Жусуповой и О.С. Салыковой опирается сугубо на качественные характеристики, что не позволяет раскрыть использование языков для решения прикладных задач и охарактеризовать их реальную производительность [2].

Тем не менее единственным способом сравнения языков программирования или их конфигураций остается создание единых условий и их типовое описание, как определенных задач, с которыми сталкивается программист при задачах разработки. Как итог, сравнение приобретает более объективный характер и позволяет говорить не о преимуществе языка и конкретной конфигурации в целом, а их вероятной большей эффективности в рамках описанного спектра задач.

Работа И.В. Родыгиной и А.В. Наливайко наглядным образом подтверждает вышеизложенное, показывая, что в задачах разработки серверной части программного приложения отдельные языки могут демонстрировать неоднозначное превосходство в одних задачах, и уступать другим языкам в других. Авторы выделяют такие критерии сравнения языков, как: скорость выдачи чистого текста, сериализация, множественные запросы к базе данных, простота использования, конфигурация, типизация и многие другие [5].

Фокусируясь на исследовании вопросов применения фреймворков для целей оптимизации производительности, заметим, что фреймворк становится одним из объективных способов компенсации недостатков конкретного языка. Исследование А.И. Федотова и Р.Г. Гильванова наглядно показывает, как фреймворк способен повлиять на итоговую производительность кода и программируемого приложения. Фреймворк предоставляет дополнительные возможности в раскрытии потенциала конкретного языка, а также компенсирует некоторые ошибки [7].

Одним из немногих исследований, посвященных сравнительному анализу различных конфигураций PHP-фреймворков, является исследование Е.С. Шевченко. Автор исследования выделяет несколько метрик для оценки фреймворков на базе языка PHP: количество обрабатываемых запросов в секунду, используемая память, время получения ответа от сервера и время на первый запуск приложения. В работе автор фокусируется на сравнении как отдельных фреймворков, так и версий языка PHP. Итоги исследования показывают, что версия PHP 7.1 демонстрирует значительный прирост производительности относительно PHP 5.6, а лучшим фреймворком по количеству обрабатываемых запросов в секунду становится Phalcon 3.1.2. Качественно автор выделяет следующий недостаток фреймворка Phalcon 3.1.2 – сложность применения среди начинающих разработчиков [8]. И хотя исследование Е.С. Шевченко вносит весомый вклад в решение задачи сравнения фреймворков для PHP, мы видим недостаток выбранных метрик и ситуаций использования PHP, что демонстрирует некоторые ограничения полученных выводов.

Важно также отметить, что сравнительный анализ фреймворков еще не раскрывался авторами в контексте оптимизации PHP. Это определяет научную новизну настоящего исследования и необходимость проведения сравнительного анализа PHP-фреймворков в контексте оптимизации производительности языка PHP.

Результаты и их обсуждение

В рамках исследования были разработаны стандартные тестовые веб-приложения для каждого из исследуемых фреймворков. Эти приложения были созданы с учетом равных условий для всех фреймворков, обеспечивая справедливое и надежное сравнение.

Все приложения были настроены для выполнения следующих задач:

1. Запись в БД: Приложение создает новую запись в базе данных. Это действие используется для измерения скорости и эффективности фреймворка при взаимодействии с базой данных на запись. Запись включает в себя операцию INSERT SQL для добавления новой записи в таблицу.

2. Чтение из БД: Приложение извлекает данные из базы данных. Этот тест используется для измерения скорости и эффективности фреймворка при взаимодействии с базой данных на чтение. Чтение включает в себя операцию SELECT SQL для получения записей из таблицы.

3. Удаление записей в БД: Приложение удаляет записи из базы данных. Этот тест используется для оценки скорости и эффективности фреймворка при выполнении операций удаления в базе данных. Удаление включает в себя операцию DELETE SQL для удаления записей из таблицы.

4. Обработка параллельных запросов: Приложение обрабатывает множество параллельных запросов. Это действие измеряет способность фреймворка эффективно обрабатывать совместную нагрузку и поддерживать высокую производительность при многопользовательских сценариях.

Все эти операции были проведены на каждом фреймворке (Laravel, Symfony, CodeIgniter, Zend, и Yii) в контролируемой среде и в идентичных условиях для каждого тестового приложения, что обеспечивает справедливость и объективность сравнительного анализа. Помимо описанных тестов, также были проведены дополнительные тесты на использование памяти и нагрузочное тестирование для дальнейшего сравнения производительности и эффективности различных фреймворков. Для более структурированного описания сравнения, сфокусируемся на различных метриках проводимого тестирования.

4.1. Время ответа на запрос страницы

Время ответа на запрос страницы – критическая метрика для любого веб-приложения, она представляет собой время между отправкой запроса клиентом и получением ответа от сервера. Наш тест включал измерение этого параметра на трех различных системах: низконагруженной, средненагруженной и высоконагруженной.

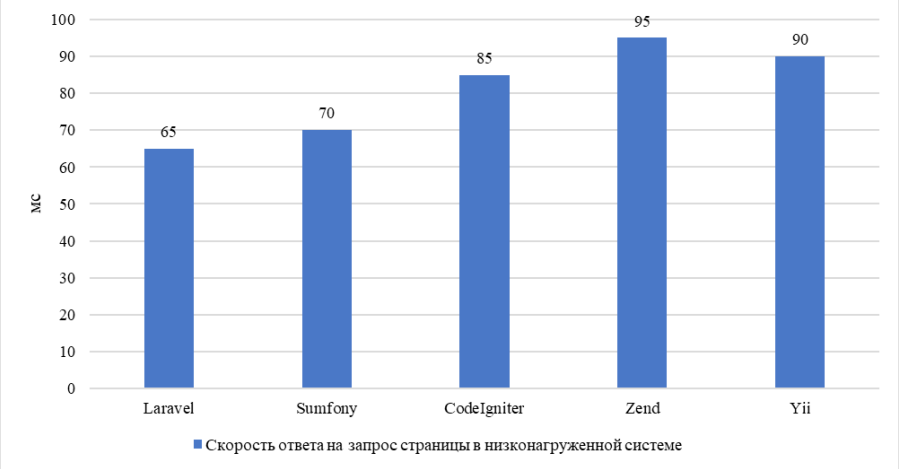

Рисунок 1 - Исследование времени ответа на запрос страницы в низконагруженной системе
DOI: https://doi.org/10.23670/IRJ.2023.137.29.1

Интерпретируя полученные результаты в структуре низконагруженной системы, выделим некоторые факторы, обосновывающие полученные данные:

1. На низконагруженной системе (Intel i5, 8GB RAM), Laravel показал лучший результат с временем ответа на запрос страницы в 65 миллисекунд. Этот результат обусловлен эффективностью Laravel в обработке входящих запросов и быстротой его внутренних механизмов, включая обработку маршрутов, представлений и баз данных.

2. Symfony, второй по быстродействию, показал время ответа 70 мс. на низконагруженной системе. Symfony является одним из самых мощных фреймворков PHP, но его сложность может немного снизить скорость обработки запросов по сравнению с Laravel.

3. CodeIgniter показал время ответа 85 мс. Этот фреймворк изначально создавался с упором на производительность, но его возраст и отсутствие некоторых современных функций, доступных в Laravel и Symfony, могут снизить его общую производительность.

4. Zend и Yii показали время ответа 95 мс и 90 мс соответственно. Они оба являются мощными и гибкими фреймворками, но их относительная сложность может увеличить время обработки запросов.

Эти результаты в основном сохраняются на средней (Intel i7, 16GB RAM) и высокой (2x Intel Xeon, 64GB RAM) системах, хотя CodeIgniter и Yii показывают небольшие улучшения при переходе на высокую систему. Это может быть связано с тем, что более мощные системы могут лучше справиться с более сложными аспектами этих фреймворков и тем самым улучшить их производительность.

4.2. Запись в базу данных

Запись в базу данных (БД) – это еще одна критическая операция для большинства веб-приложений. Эффективность этой операции напрямую влияет на общую производительность приложения, особенно в системах, где выполняется большое количество записей.

В нашем исследовании мы измеряли время, необходимое каждому фреймворку для записи в БД, используя стандартную операцию SQL INSERT на всех трех системах. Замерялась средняя продолжительность операции, включающая время подготовки запроса, его выполнения и получения ответа от БД.

Сравнение среднего времени продолжительности операции сгруппировано на рис. 2:

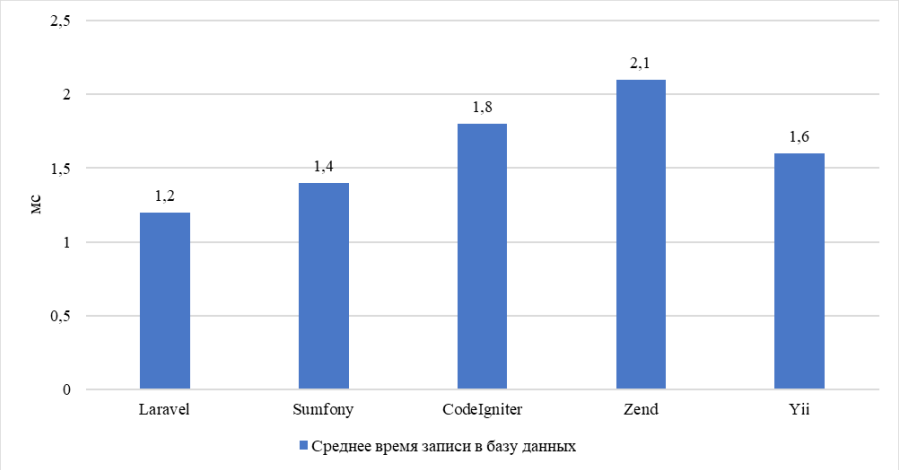

Рисунок 2 - Среднее время записи в базу данных
DOI: https://doi.org/10.23670/IRJ.2023.137.29.2

Опираясь на полученные результаты, сформируем их дополнительное описание:

1. Laravel, с средним временем записи в БД около 1.2 мс, показал наилучшие результаты. Это обусловлено встроенными функциями оптимизации Laravel для работы с БД, включая эффективное кеширование запросов и оптимизированный ORM (Object-Relational Mapping).

2. Symfony занял второе место со средним временем записи в БД 1.4 мс. Symfony использует Doctrine, один из самых мощных ORM в PHP, который, хотя и обеспечивает большую гибкость, может быть немного медленнее в некоторых операциях, включая запись в БД.

3. Yii и CodeIgniter показали среднее время записи в БД 1.6 мс. и 1.8 мс. соответственно. Оба фреймворка обладают простыми и эффективными механизмами работы с БД, но их возраст и отсутствие некоторых современных функций могут увеличить время выполнения операций записи.

4. Zend со средним временем записи в БД 2.1 мс. показал самые низкие результаты. Zend Framework является одним из самых мощных и гибких, но его сложность и множество вариантов конфигурации могут привести к более длительному времени выполнения запросов к БД.

Очевидно, что эффективность операций записи в БД существенно влияет на общую производительность веб-приложения. Исходя из проведенного исследования, Laravel демонстрирует наиболее высокую производительность при выполнении таких операций.

4.3. Чтение из базы данных

Чтение из базы данных (БД) является не менее значимой функцией на базе веб-приложения, поскольку позволяет извлекать записанную информацию. В нашем исследовании мы измеряли время, которое требуется каждому фреймворку для выполнения стандартного SQL SELECT запроса (чтение базы данных). Время измерялось от момента отправки запроса до момента получения ответа.

Результаты сравнения скорости осуществления операций сгруппированы на рис. 3:

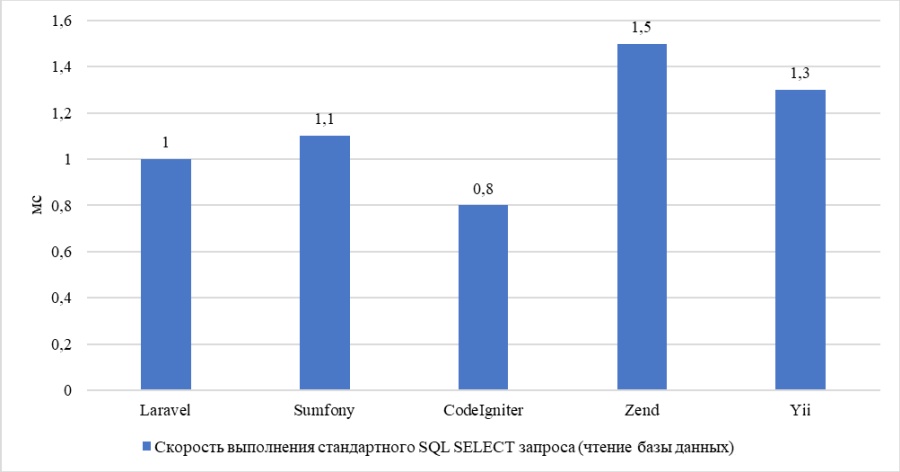

Рисунок 3 - Скорость выполнения стандартного SQL SELECT запроса

DOI: https://doi.org/10.23670/IRJ.2023.137.29.3

Опираясь на полученные результаты, сформируем их дополнительное описание:

1. CodeIgniter, с средним временем чтения из БД 0.8 мс., показал наилучшие результаты. Этот легкий фреймворк всегда был известен своей высокой производительностью, и наше исследование подтвердило эту репутацию в контексте операций чтения из БД.

2. Laravel, показавший среднее время чтения из БД 1 мс., занял второе место. Это отражает эффективность встроенных в Laravel средств работы с БД и их способность быстро обрабатывать запросы на чтение.

3. Symfony со средним временем чтения из БД 1.1 мс., следует за Laravel. Doctrine, используемый Symfony для работы с БД, хотя и является мощным ORM, может быть немного медленнее по сравнению с другими из-за своей сложности и гибкости.

4. Yii и Zend показали среднее время чтения из БД 1.3 мс. и 1.5 мс. соответственно. Оба фреймворка предлагают сложные и мощные инструменты для работы с БД, но этот функционал может стать причиной увеличения времени выполнения запросов на чтение.

Опираясь на полученные результаты, следует отметить, что разница во времени чтения из баз данных между всеми фреймворками не значительна, что говорит о идентичном использовании достаточно эффективных механизмов для выполнения этой операции. Однако даже небольшие различия могут демонстрировать накопительный эффект и в конечном счете оказывать влияние на общую производительность приложения, особенно при больших объемах данных и высокой нагрузке.

4.4. Параллельные запросы

Параллельные запросы – это запросы, которые идут одновременно, а не последовательно. Скорость их исполнения достаточно важна в современных веб-приложениях, где сотни или тысячи пользователей могут одновременно обращаться к приложению и реализовывать схожие запросы. Важность умения эффективно обрабатывать

параллельные запросы не может быть недооценена: это может стать ключевым фактором при поддержании высокой производительности приложения в условиях высокой нагрузки.

Под параллельными запросами понимаются ситуации, когда одновременно происходит множество операций, таких как чтение или запись в БД, обработка HTTP-запросов или выполнение бизнес-логики. В качестве примера можно привести ситуацию, когда одновременно идут операции чтения и записи в БД, когда одни пользователи вносят изменения в данные, в то время как другие читают их, или когда множество пользователей одновременно пытаются получить доступ к одной и той же странице веб-приложения.

В рамках нашего исследования, мы оценивали способность каждого из фреймворков обрабатывать большое количество параллельных запросов. Это требует ориентации на высоконагруженную систему. Результаты сравнения показаны на рис. 4:

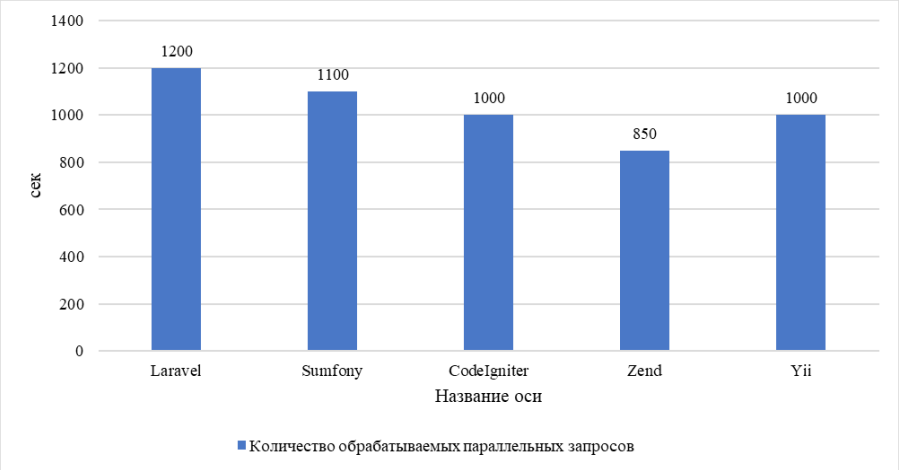

Рисунок 4 - Количество обрабатываемых параллельных запросов в секунду
DOI: https://doi.org/10.23670/IRJ.2023.137.29.4

Интерпретируя результаты сравнения, представленного на рис. 4, сформируем качественное описание изменений:

1. Laravel и Symfony показали наилучшую производительность, обрабатывая до 1200 и 1100 запросов в секунду соответственно на высоконагруженной системе. Это свидетельствует о том, что эти фреймворки обладают отличными механизмами управления ресурсами и могут эффективно работать в условиях высокой нагрузки.

2. CodeIgniter и Yii также показали хорошие результаты, обрабатывая около 1000 запросов в секунду. Эти фреймворки, хотя и не демонстрируют такой высокой производительности, как Laravel или Symfony, все же достаточно эффективно справляются с обработкой большого числа одновременных запросов.

3. Zend, обрабатывающий около 850 запросов в секунду, показал наименьшую производительность. Этот фреймворк обладает мощными и гибкими функциональными возможностями, но, возможно, ему не хватает эффективности в условиях высокой нагрузки и большого числа параллельных запросов.

Таким образом, в ситуации, когда веб-приложение призвано обрабатывать множество одновременных запросов, следует уделить внимание выбору соответствующего фреймворка. Лидерство в данном вопросе демонстрируют Laravel и Symfony.

4.5. Использование памяти

Использование памяти – это общий объем, который будет занимать приложение на сервере. Объем памяти, который требуется для работы веб-приложения, является ключевым параметром, поскольку он напрямую коррелирует с общей эффективностью работы и способностью масштабироваться. Если приложение неэффективно использует память, это может негативно сказаться на производительности и даже вызвать сбои системы при высокой нагрузке. Таким образом, при выборе технологий и инструментов следует отдать предпочтение тем, что обеспечивают рациональное использование памяти.

Существует множество сценариев использования памяти веб-приложениями. Рассмотрим несколько прикладных примеров их воспроизводства:

Пример 1: установим, что осуществляется управление популярным новостным сайтом, который каждый день посещает несколько миллионов человек. В день крупного новостного события число посещений может внезапно увеличиться в разы. Если веб-сайт не способен эффективно использовать память, это может вызвать замедление работы сайта или даже его полное «падение» из-за перегрузки системы.

Пример 2: возьмем для примера веб-приложение для обработки изображений. Если приложение использует большой объем памяти для обработки каждого изображения, его производительность может снизиться, и это приведет к долгому времени ожидания для пользователей, особенно при работе с большими изображениями или при выполнении сложных операций обработки.

Пример 3: допустим, осуществляется разработка SaaS-решения для управления проектами, которое будут использовать множество пользователей одновременно. Если каждый пользовательский сеанс потребует большого объема памяти, это может ограничить количество пользователей, которые могут работать с приложением одновременно, и может требовать дополнительных инвестиций в серверные ресурсы, чтобы обслуживать все запросы.

Во всех этих примерах рациональное использование памяти может значительно усилить производительность и масштабируемость приложения, улучшая при этом общий пользовательский опыт и снижая затраты на серверные ресурсы. В рамках настоящего исследования мы измеряли потребление памяти каждым из фреймворков при обработке типичного запроса. Замеры проводились на разных системах, начиная с низконагруженной и заканчивая высоконагруженной. Результаты исследования представлены на рис. 5:

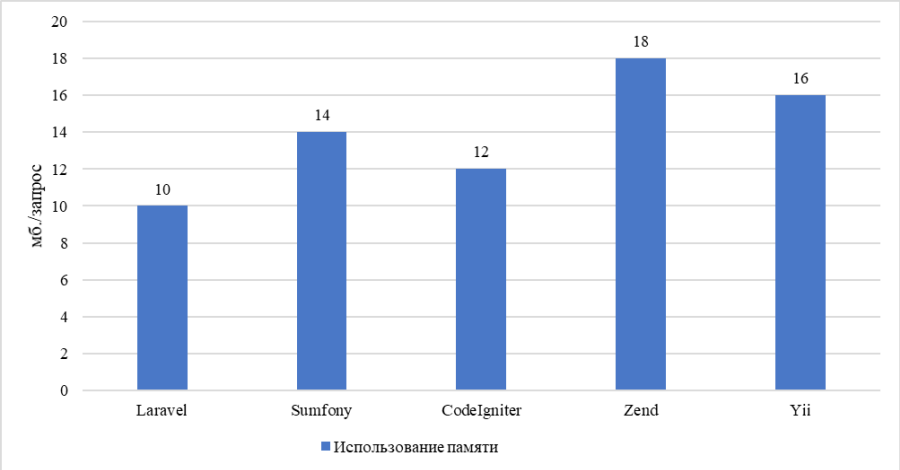

Рисунок 5 - Использование памяти
DOI: https://doi.org/10.23670/IRJ.2023.137.29.5

Результаты сравнения позволяют выделить следующие итоги:

1. Laravel, потребляя в среднем около 10 МБ памяти на запрос, показал наиболее эффективное использование памяти. Это свидетельствует о хорошем уровне оптимизации фреймворка и его способности обрабатывать запросы, минимизируя при этом использование ресурсов.

2. CodeIgniter, с потреблением около 12 МБ памяти на запрос, занял второе место. Этот фреймворк также известен своим небольшим потреблением ресурсов, и эти результаты подтверждают эту репутацию.

3. Symfony, Yii и Zend показали потребление памяти около 14 МБ, 16 МБ и 18 МБ на запрос соответственно. Это может быть связано с тем, что данные фреймворки предоставляют более широкий спектр функциональности, что может требовать большего количества ресурсов.

Следует отметить, что эффективное использование памяти не всегда означает высокую производительность веб-приложения. Более важным является баланс между потреблением ресурсов и предоставляемой функциональностью, и именно этот баланс следует искать при выборе фреймворка для конкретной задачи.

4.6. Нагрузочное тестирование

В ходе нагрузочного тестирования была проведена оценка производительности различных фреймворков, а именно Laravel, Symfony, CodeIgniter, Yii и Zend, на высоконагруженной системе. Целью тестирования было определение, какие из этих фреймворков лучше всего справляются с обработкой большого количества запросов в секунду. Сравнение результатов сгруппировано на рис. 6:

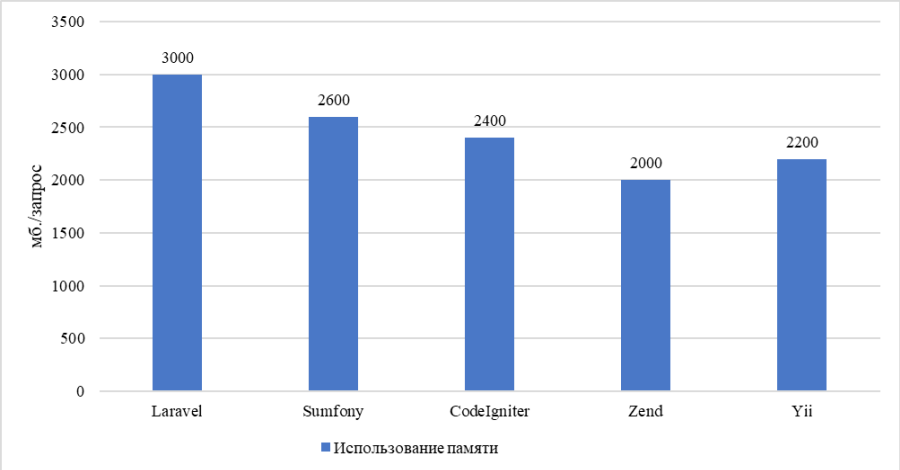

Рисунок 6 - Результаты нагрузочного тестирования
DOI: https://doi.org/10.23670/IRJ.2023.137.29.6

Примечание: обработка запросов в секунду

Результаты тестирования показали, что:

1. Laravel продемонстрировал наилучшую производительность, обрабатывая до 3000 запросов в секунду. Это означает, что веб-приложение, построенное на Laravel, способно эффективно обслуживать большое количество одновременных запросов, что особенно важно в условиях высоконагруженных систем.

2. Symfony показал производительность в 2600 запросов в секунду, что также является впечатляющим результатом. Этот фреймворк обладает мощными инструментами и возможностями, которые позволяют обрабатывать большое количество запросов эффективно и без снижения производительности.

3. CodeIgniter продемонстрировал производительность в 2400 запросов в секунду. Это говорит о том, что этот фреймворк также способен эффективно обрабатывать высокую нагрузку и предоставлять отзывчивый опыт пользователям.

4. Yii продемонстрировал производительность в 2200 запросов в секунду. Этот фреймворк обладает высокой скоростью выполнения и отличной производительностью, что делает его привлекательным вариантом для приложений, требующих высокой отзывчивости.

5. Zend показал производительность в 2000 запросов в секунду. В ходе тестирования он продемонстрировал способность обрабатывать значительный объем запросов, но немного уступает другим фреймворкам в данном сравнении.

Производительность фреймворков при обработке запросов в секунду имеет важное значение для веб-приложений, таких как онлайн-магазины, социальные сети, финансовые сервисы и другие, где требуется одновременное взаимодействие множества пользователей с системой. Быстрая и эффективная обработка запросов обеспечивает плавный и отзывчивый пользовательский опыт, минимизирует задержки и улучшает общую производительность приложения.

Заключение

На основе проведенного сравнительного анализа можно сделать вывод, что Laravel является наиболее производительным PHP-фреймворком, с учетом всех рассмотренных метрик. Однако это не означает, что другие фреймворки не могут быть подходящими для определенных задач разработчика. Например, CodeIgniter показал лучшую производительность при чтении из базы данных, что может быть критично для приложений, в которых это действие является наиболее частым. Важно адаптивно подходить к выбору конкретного фреймворка, принимая во внимание подобные различия.

Не менее значимой является необходимость учета того, что на производительность фреймворка могут влиять многие факторы, включая качество кода, структуру приложения, настройки сервера, и многое другое. Поэтому этот анализ должен рассматриваться как отправная точка при выборе подходящего PHP-фреймворка для проекта.

Наконец, следует учесть, что на протяжении последних нескольких лет Laravel непрерывно развивается и улучшается, что подтверждается его высокими показателями в данном исследовании. В то же время Symfony и Yii также демонстрируют уверенные результаты и имеют свои преимущества, что делает их подходящими кандидатами для определенных типов проектов.

Практическая значимость полученных результатов исследования заключается в возможностях их использования при выборе необходимого фреймворка для разработки веб-приложений. Результаты сравнительного анализа могут стать основанием для выбора фреймворка под потребности разработчика, учитывая полученные данные о скорости выполнения запросов определенного типа. В глобальном представлении, исследование вносит определенный положительный вклад в решение вопросов оптимизации и повышения производительности разработки на PHP.

Конфликт интересов

Не указан.

Рецензия

Сообщество рецензентов Международного научно-исследовательского журнала
DOI: https://doi.org/10.23670/IRJ.2023.137.29.7

Conflict of Interest

None declared.

Review

International Research Journal Reviewers Community
DOI: https://doi.org/10.23670/IRJ.2023.137.29.7

Список литературы / References

1. Гребенникова Е.К. Оптимизация времени работы алгоритма для многоядерных процессоров в программном комплексе теплового проектирования электронных модулей / Е.К. Гребенникова, П.И. Кандалов // ВК. — 2018. — №3 (31). — с. 188-194.

2. Давыдов А.В. Сравнение различных языков программирования, применяемых в машинном обучении / А.В. Давыдов, А.К. Жусупова, О.С. Салыкова // Вестник науки. — 2023. — №2 (59). — с. 155-165.

3. Егоров М.Ю. Применение графических ускорителей для повышения производительности вычислений при численном моделировании функционирования сложных технических систем / М.Ю. Егоров, С.М. Егоров, Д.М. Егоров // Вестник ПНИПУ. Аэрокосмическая техника. — 2015. — №1 (40). — с. 81-91.

4. Кочнев А.А. Web Development с использованием PHP и фреймворка Laravel / А.А. Кочнев // EESJ. — 2023. — №1-1 (86). — с. 4-11.

5. Родыгина И.В. Сравнительный анализ технологий для разработки серверной части системы управления продажами / И.В. Родыгина, А.В. Наливайко // Известия ЮФУ. Технические науки. — 2021. — №4 (221). — с. 256-266.

6. Рокотянская В.В. Исследование webassembly и сравнение производительности с javascript / В.В. Рокотянская, В.С. Абрамов // Вестник АГТУ. Серия: Управление, вычислительная техника и информатика. — 2023. — №2. — с. 93-100.

7. Федотова А.И. Разработка кросс-платформенных приложений на языке Python и фреймворке Kivy / А.И. Федотова, Р.Г. Гильванов // Интеллектуальные технологии на транспорте. — 2022. — №2 (30). — с. 53-58.

8. Шевченко Е.С. Сравнительное тестирование PHP-фреймворков / Е.С. Шевченко // Вестник науки и образования. — 2019. — №10-2 (64). — с. 40-45.

9. Laaziri M. A Comparative Study of PHP Frameworks Performance / M. Laaziri, K. Benmoussa, S. Khoulji [et al.] // Procedia Manufacturing. — 2019. — № 32. — p. 864-871.

10. Ramirez E. Pro PHP Application Performance Tuning PHP Web Projects for Maximum Performance / E. Ramirez. — URL: https://www.academia.edu/11399359/Pro_PHP_Application_Performance_Tuning_PHP_Web_Projects_for_Maximum_Performance (accessed: 15.06.2023)

Список литературы на английском языке / References in English

1. Grebennikova E.K. Optimizacija vremeni raboty algoritma dlja mnogojadernyh processorov v programmnom komplekse teplovogo proektirovanija jelektronnyh module [Optimization of the Running Time of the Algorithm for Multi-core Processors in the Software Package for the Thermal Design of Electronic Modules] / E.K. Grebennikova, P.I. Kandalov // VK. — 2018. — №3 (31). — p. 188-194. [in Russian]

2. Davydov A.V. Sravnenie razlichnyh jazykov programmirovanija, primenjaemyh v mashinnom obuchenii [Comparison of Various Programming Languages Used in Machine Learning] / A.V. Davydov, A.K. Zhusupova, O.S. Salykova // Vestnik nauki [Bulletin of Science]. — 2023. — №2 (59). — p. 155-165. [in Russian]

3. Egorov M.Ju. Primenenie graficheskih uskoritelej dlja povyshenija proizvoditel'nosti vychislenij pri chislennom modelirovanii funkcionirovanija slozhnyh tehnicheskih system [The Use of Graphics Accelerators to Improve the Performance of Calculations in Numerical Simulation of the Functioning of Complex Technical Systems] / M.Ju. Egorov, S.M. Egorov, D.M. Egorov // Bestnik PNIPU. Ajerokosmicheskaja tehnika [Bulletin of PNRPU. Aerospace Engineering]. — 2015. — №1 (40). — p. 81-91. [in Russian]

4. Kochnev A.A. Web Development s ispol'zovaniem PHP i frejmvorka Laravel [Web Development Using PHP and the Laravel Framework] / A.A. Kochnev // EESJ [EESJ]. — 2023. — №1-1 (86). — p. 4-11. [in Russian]

5. Rodygina I.V. Sravnitel'nyj analiz tehnologij dlja razrabotki servernoj chasti sistemy upravlenija prodazhami [Comparative Analysis of Technologies for the Development of the Server Part of the Sales Management System] / I.V. Rodygina, A.V. Nalivajko // Izvestija JuFU. Tehnicheskie nauki [News of the Southern Federal University. Technical Science]. — 2021. — №4 (221). — p. 256-266. [in Russian]

6. Rokotjanskaja V.V. Issledovanie webassembly i sravnenie proizvoditel'nosti s javascript [Webassembly Research and Performance Comparison with Javascript] / V.V. Rokotjanskaja, V.S. Abramov // Vestnik AGTU. Serija: Upravlenie, vychislitel'naja tehnika i informatika [Vestnik ASTU. Series: Management, Computer Engineering and Informatics]. — 2023. — №2. — p. 93-100. [in Russian]

7. Fedotova A.I. Razrabotka kross-platformennyh prilozhenij na jazyke Python i frejmvorke Kivy [Development of Cross-platform Applications in Python and the Kivy Framework] / A.I. Fedotova, R.G. Gil'vanov // Intellektual'nye tehnologii na transporte [Intelligent Technologies in Transport]. — 2022. — №2 (30). — p. 53-58. [in Russian]

8. Shevchenko E.S. Sravnitel'noe testirovanie PHP-frejmvorkov [Comparative Testing of PHP Frameworks] / E.S. Shevchenko // Vestnik nauki i obrazovanija [Bulletin of Science and Education]. — 2019. — №10-2 (64). — p. 40-45. [in Russian]

9. Laaziri M. A Comparative Study of PHP Frameworks Performance / M. Laaziri, K. Benmoussa, S. Khoulji [et al.] // Procedia Manufacturing. — 2019. — № 32. — p. 864-871.

10. Ramirez E. Pro PHP Application Performance Tuning PHP Web Projects for Maximum Performance / E. Ramirez. — URL: https://www.academia.edu/11399359/Pro_PHP_Application_Performance_Tuning_PHP_Web_Projects_for_Maximum_Performance (accessed: 15.06.2023)