https://doi.org/10.31891/2307-5732-2023-317-1-155-161
УДК 004.9

ПАЗДРІЙ Ігор
Західноукраїнський національний університет, м. Тернопіль, Україна
https://orcid.org/0000-0001-8505-4128
e-mail: pir@wunu.edu.ua

ПОРІВНЯЛЬНИЙ АНАЛІЗ СИСТЕМ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА ОСНОВІ ФРЕЙМВОРКІВ

У статті проведено порівняльний аналіз веб-фреймворків на основі мови програмування Python. Правильний вибір інструменту розробки програмного забезпечення дозволить розробникам пришвидшити процес проектування та програмної реалізації веб-ресурсів.

З розвитком мов програмування почали з'являтися рішення, які полегшують програмістам створення веб-додатків. За останні роки було розроблено багато інструментів цього типу, зокрема веб-фреймворків. Важливим критерієм оцінки веб-фреймворка є час розробки базових елементів, таких як сторінка реєстрації та додаткових елементів – специфічних для конкретного випадку. Автором проведено аналіз продуктивності фреймворків за критеріями часу написання програмних модулів та об'єму програмного коду. Від вибору правильного веб-фреймворка залежить якість, надійність та час реалізації кожного розроблюваного програмного засобу.

Існуючі веб-фреймворки відрізняються один від одного і вибір їх може стати для програміста складним завданням. Тому потрібно навести кілька параметрів, які є загальними для фреймворків, згідно яких слід вибирати потрібний веб-фреймворк. Вибрані критерії дозволяють якісніше та об'єктивніше оцінити фреймворки та пришвидшити процес проєктування та програмної реалізації веб-ресурсів. Через високий попит на веб-додатки розробники повинні розробляти економічно ефективні, безпечні та добре кодовані веб-додатки.

Ключові слова: фреймворки, веб-ресурси, Flask, Django.

PAZDRIY Ihor
West Ukrainian National University, Ternopil, Ukraine.

COMPARATIVE ANALYSIS OF SOFTWARE DEVELOPMENT SYSTEMS BASED ON FRAMEWORKS

The article provides a comparative analysis of web frameworks based on the Python programming language. The right choice of a software development tool will allow developers to speed up the process of designing and software implementation of web resources. With the development of programming languages, solutions have begun to appear that make it easier for programmers to create web applications. In recent years, many tools of this type have been developed, including web frameworks. Some of them are used to develop small web applications, and other frameworks are used to create large-scale systems.

The existing web frameworks differ from each other, but choosing them can be a difficult task for a programmer. Therefore, it is necessary to specify several parameters that are common to frameworks, according to which the desired web framework should be selected. The selected criteria make it possible to evaluate the frameworks more qualitatively and more objectively and speed up the process of design and software implementation of web resources. Due to the high demand for web applications, developers need to develop cost-effective, secure and well-coded web applications. Analyzing the Django framework, it can be argued that it is the most popular for web development, written in the Python programming language and using the MVC architecture. It is ideal for application projects with limited time and a small budget. If the application is not very voluminous and only simple URL routing and templates with a simple context are needed, then the Flask framework can be used instead of Django.

The framework for building a Flask application in the resulting HTML files contains additional tags and scripts. The mechanism of work of processing forms also works differently. Flask stores more information about form fields on pages. The size of HTML files created using the Flask framework turns out to be significantly larger.

***Keywords**: frameworks, web resources, Flask, Django.*

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Одним з важливих завдань при створенні веб-додатків є написання програмного коду легкого для читання, зрозумілого для інших програмістів та для багаторазового використання. Цю практику часто називають принципом DRY (Don't repeat yourself). Він стверджує, що код повинен створюватися таким чином, щоб жоден його фрагмент не повторювався в різних частинах програми.

Повторювані дії повинні виконуватися генераторами коду або сценаріями, призначеними для автоматизації роботи. Зі збільшенням вимог користувачів та рівня складності веб-додатків виникла необхідність створення інструментів для підтримки процесу розробки програмного забезпечення.

Ще одна складність для будь-якого програміста – розуміння коду написаного іншими програмістами. Якщо таке повторюватиметься у багатьох місцях програми, то слідкування за певним фрагментом коду може стати досить складним завданням. Це призводить до неоднозначних інтерпретацій та підвищує складність програми. Це обмежує можливість розширення існуючих додатків, а також ускладнює передачу роботи над програмним забезпеченням іншим розробникам, які не брали участі у попередніх проєктах.

Вирішенням вищезазначених проблем стали фреймворки.

Фреймворк – це готова форма моделі в IT, яка повторно використовується для програмної платформи, що включає також дизайн високого рівня. Це структура викладена на мові програмування, у той

час як інша техніка повторного використання залежить від спеціального призначення і потребує спеціальних програмних засобів. Це шаблон для програмної платформи, на основі якого можна дописати власний програмний код [1]. Структура зосереджена на кількох основних критеріях, що характеризують процес розробки програмного забезпечення, зокрема: продуктивність, здатність до модифікування програми, функціональність та відповідність стандартам [2].

Фреймворки надають програмістам певні функціонали та готові модулі, що дозволяють зосередитися лише на найважливіших аспектах розробки програмного забезпечення. Рутинні завдання, такі як обробка сеансу або перевірка даних у формах, мають другорядне значення, оскільки вони забезпечуються належним чином налаштованим каркасом програми. Налаштувавши функціональну групу під свої потреби та врахувавши бізнес-логіку, програміст може приступати до роботи. Саме це робить фреймворки додатків популярними в середовищі розробки, оскільки розробникам не доводиться "винаходити колесо" з кожним новим проєктом.

Слід зазначити, що фреймворки розробки додатків забезпечують не тільки готові сценарії чи бібліотеки, а й умови програмування, які значною мірою впливають на мінімізацію часу, необхідного для створення програмного забезпечення високої якості. Деякі з них використовують для розробки невеликих веб-додатків, інші – для створення масштабних систем, які потребують численних рішень та функціональних можливостей. Кожен програміст, якому доводиться вибирати новий робочий інструмент, повинен спочатку визначитися із його вибором.

Сучасні фреймворки володіють низкою певних переваг та недоліків, а тому актуальним є питання вибору правильного та відповідного фреймворку або їх набору для кожного конкретного проекту, адже від правильного вибору залежить якість, час реалізації та надійність кожного розроблюваного програмного продукту.

Об'єкт дослідження – фреймворки для розробки веб-систем.

Предмет дослідження – підходи до розробки програмного забезпечення.

Метою даної роботи є проведення порівняльного аналізу вибраних фреймворків для створення веб-додатків. Для цього слід здійснити вибір, описати та обґрунтувати критерії порівняння фреймворків. Тематичне дослідження послужить вичерпною темою практичного аспекту використання вибраних фреймворків для створення програм. Результати дослідження визначають доцільність вибору та застосування сучасних фреймворків для розробки програмного забезпечення.

Для досягнення поставленої мети визначено такі основні завдання дослідження:

- виділити сучасні фреймворки на основі мови програмування Python;
- виділити критерії оцінки продуктивності фреймворків;
- на основі аналітичного підходу здійснити порівняльний аналіз фреймворків веб-розробки на основі визначених критеріїв.

Аналіз досліджень та публікацій

З розвитком веб-технологій також розвиваються інтернет-сторінки. Розвиток Інтернету робить веб-додатки надзвичайно важливими. Сторінки статичні були замінені на динамічні сторінки з великою кількістю інтерактивних елементів, які можна змінювати багато разів за досить короткий час. Для допомоги створення таких веб-сторінок розробникам потрібні фреймворки. Існує велика кількість різних фреймворків. Науковці в сфері веб-технологій приділяють значну увагу сучасним фреймворкам: їх опису, аналітичному аналізу, порівнянню.

У роботі [3] автори наводять результати порівняльного аналізу трьох фреймворків для створення веб-додатків Express, Hapi і Koa. В рамках роботи реалізовано серверні тестові додатки, які виконують типові операції з базою даних. На основі отриманих результатів виявилося, що фреймворки Hapi і Koa, які працюють над одним об'єктом або рядком Hello World досягли найкращого часу відповіді. Koa виявився найкращим при більших навантаженнях, досягаючи часу відповіді приблизно на 20% кращого, ніж Express. При високих навантаженнях Hapi виявився гіршим, досягнувши більш ніж у 2 рази довшого часу відгуку, ніж фреймворк Koa.

Автори роботи [4] реалізували модель архітектури MVC (Model View Controller) із фреймворками Laravel і Slim. Шаблони проектування MVC є добре відомими шаблонами, які використовуються для інтерактивних архітектур програмного забезпечення. Проведений порівняльний аналіз продуктивності фреймворків Laravel і Slim під час навантажувального тестування на сторінці інформаційної панелі за допомогою Apache JMeter. Результати, отримані в від порівняння продуктивності, свідчать про те, що фреймворк Slim швидший і кращий за фреймворк Laravel.

Порівняння переваг та недоліків веб-розробки за допомогою фреймворку з розробкою в стилі бібліотеки розглянуто у роботі [5]. Це порівняння базується на двох основних показниках: вплив на керівництво та вплив на розробника. Багаторазове використання, гнучкість, економічність та рентабельність – це критерії, які враховуються керівництвом у визначенні свого бізнес-процесу. За вибір методу, бібліотеки чи фреймворку відповідає компанія. Фреймворк допомагає розробнику автоматично створювати структуру мережі. Показано, що обидва методи як фреймворк, так і бібліотека, з точки зору управління компанією, можуть використовуватися багаторазово.

У статті [6] представлено порівняння продуктивності реляційних баз даних SQL Server, MySQL і

PostgreSQL з використанням фреймворку Laravel. Оцінено час виконання для різних типів запитів, як простих, так і з використанням конкатенації стовпців і таблиць. Отримані результати для одних і тих самих структур баз даних відрізнялися кількістю операцій. Розглядаючи всі проведені дослідження, можна зробити висновок, що у випадку баз даних з не надто великою кількістю записів (до 1000 записів), а технічні параметри пристрою, на якому працює база даних, низького або середнього класу, MySQL працює досить добре.

Автор роботи [7] показав, що за допомогою веб-фреймворку CodeIgniter можна досить швидко створити невеликий проект, CakePHP – дозволить швидко написати проект, але оптимізувати його буде складно, Symfony – ідеально підходить під середні та великі проекти.

У статті [8] розглянуто та порівняно функції безпеки трьох найпопулярніших і найпотужніших фреймворків PHP Laravel, CodeIgniter і Symfony. Виходячи з результатів порівняння Laravel найбільш захищений. У цій статті наведено опис запропонованої концепції для впровадження безпечних програм з використанням фреймворків.

Метою статті [9] є порівняння продуктивності популярних фреймворків Angular і Vue.js у контексті вибору кращого. З наведених у статті результатів автори зробили висновок, що фреймворк Angular є більш складним, але завдяки жорсткому набору тексту він досить ефективний для створення великих проектів. Vue.js має менше функцій і, здається, є кращим вибором для програмістів-початківців, але одночасно – це більш гнучкий фреймворк, ніж Angular, і його можна використовувати для написання як великих, так і малих проектів. Незважаючи на те, що TypeScript, на якому написаний Angular, складніший і важчий для вивчення, розробити та підтримувати програмне забезпечення, створене з його допомогою, простіше і ефективніше. Фреймворк Vue.js є кращим програмним середовищем, ніж фреймворк Angular, для створення веб-додатків середнього рівня складності.

У статті [10] автор представив аналіз продуктивності восьми фреймворків. Проведене дослідження полягало у вимірюванні часу створення, модифікації та видалення елементів HTML на веб-сайті. Порівняння результатів цих досліджень показує, котрий з фреймворків найшвидший. Порівнюючи середні значення часу, Vue.js швидший, ніж Angular, але тільки у випадку видалення елементів HTML. Автор робить висновок, що Angular має найкращі результати та займає перше місце майже у всіх проведених тестах. Vue.js є одним із найшвидших фреймворків для створення та модифікації елементів, хоча він все ще повільніший за Angular у виконанні такого типу операцій.

Автор статті [11] порівнює структури для створення веб-сайту в системі керування вмістом Sitkore. Щоб визначити найбільш прийнятну технологію, автор порівнює такі показники, як кількість рядків вихідного коду, обсяг дискового простору, який використовується виконуваними файлами, складність кодування, архітектурні обмеження, введені фреймворком і продуктивність. На основі цього дослідження зроблено висновок, що для створення такого типу додатків найкраще використовувати бібліотеку React. Автор також зазначає, що фреймворк Vue.js має великі перспективи витіснити React завдяки легкому входу, мінімальній базовій конфігурації та простоті розширення архітектури коду. Для складного та багатофункціонального проекту Angular є головним конкурентом React завдяки гнучкості коду та швидкості відтворення сторінок.

Автори статті [12] провели порівняння фреймворків Angular, Vue.js і React, щоб визначити, який з них ефективніший для створення додатків SPA, для створення додатків типу MPA або для створення обох типів цих додатків. Вибрані фреймворки досліджувались на здатність генерувати сторінку на стороні сервера, здатність писати та керувати великою кількістю складного коду та можливості фреймворків у разі стиснення та мінімізації вихідних файлів. У результаті дослідження зроблено висновок, що Vue.js є найбільш придатним для створення MPA, а Angular є гіршим вибором через його розмір і складність, а також його неспроможність контролю невеликих частин структури DOM. У випадку SPA Angular, на думку авторів, це найкращий вибір. Vue.js є ефективним вибором як для розробки SPA, так і для MPA.

Виклад основного матеріалу

Параметри оцінювання фреймворків.

Врахувавши, що фреймворки – це інструмент полегшення та покращення створення і запуску веб-додатків, для програміста відпадає необхідність самостійно прописувати велику кількість коду та затрачати час на пошук можливих помилок та прорахунків. На сьогоднішній час існує можливість вибору відповідних фреймворків веб-програмування. Вибір фрейморку здійснюється, в залежності від поставленого завдання.

Слід провести вибір критеріїв порівняння фреймворків. На основі цих критеріїв були проведені вимірювання для кожної з аналізованих структур створення веб-додатків.

Одним з таких параметрів є час, витрачений на розробку додатків. З точки зору менеджера команди розробників, досить важливо знати орієнтовний час, який слід витратити на розробку. Чим більше часу займе створення програми, тим дорожчим буде проект. Без таких даних легко недооцінити час реалізації та бюджет цілого проекту. Тому час, витрачений на розробку додатків є одним із найважливіших порівняльних критеріїв. Вимірювання включають час, витрачений на створення компонентів програми та її відлагодження. Також було виміряно середній час роботи над функційними можливостями розробленого веб-додатку.

Наступним досліджуваним параметром є час, витрачений на зміну програми. Знання часу,

витраченого на розробку програми, є важливим аспектом, але слід також враховувати можливість її модифікації. Час, витрачений на зміну існуючої програми, включаючи додавання, видалення та зміну функційних можливостей, показує якість внутрішньої структури фірми створення веб-додатків. Цей показник також дає уявлення про те, наскільки коштовною може бути підтримка програми в майбутньому. Вимірювання включають час, витрачений на зміну програми, а також кількість файлів, які підлягають зміні при виконанні певного завдання.

Наступним порівняльним критерієм є розмір написаного коду. Цей критерій пов'язаний з першим критерієм. Він допомагає оцінити загальний час, витрачений на створення веб - програми. Якщо результати вимірювань, в певній мірі, відрізняються від результатів першого критерію, то це може означати, що робота з використанням конкретного фреймворку є більш складною. Цей показник є важливим для початківців створення веб-додатків. Час, необхідний для підвищення продуктивності розробника, може суттєво збільшуватися. Цей критерій також може вказувати на перевагу використання компонентної платформи розробки додатків, оскільки, як очікується, повторне використання коду буде на значно вищому рівні. Вимірювання поділялись на розміри збереженої конфігурації, моделі та шаблонів. Результати містять вимірювання розміру (у байтах) усіх файлів використаних для реалізації даної функційності та середнього розміру файлу для кожного рішення.

Розмір згенерованого коду для веб-додатків залишається важливою проблемою. Менший код програми, зазвичай, означає менший час очікування завантаження веб-браузера. Користувачі не завжди люблять довго чекати на відповідь від сервера. Якщо час очікування буде постійно надто довгим, вони почнуть шукати альтернативні рішення.

Кожний фреймворк має певні переваги для керування шаблонами сторінок та загальною структурою файлів. Отриманий розмір HTML -файлів також може відрізнятися, оскільки вони можуть містити код з різних мов сценаріїв. Використані шаблони допомагають упорядкувати код та відокремити бізнес-логіку від рівня презентації. Розмір сформованого коду веб-програми вимірювався в байтах на шаблонах HTML. Результати вимірювань також включають середній розмір створених шаблонів.

Порівняльний аналіз сучасних фреймворків

Для порівняльного аналізу вибрано два сучасних фреймворки Django та Flask. Структуру веб-фреймворку Django наведено на рисунку 1.

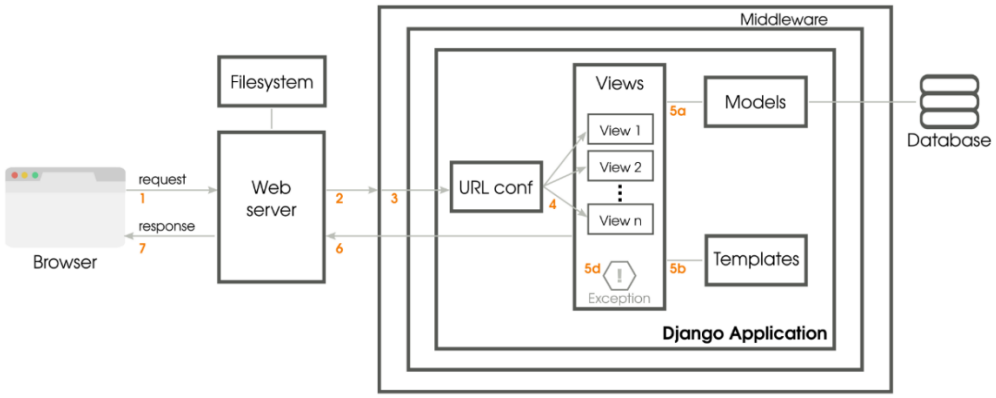

Рис. 1. Структура веб-фреймворку Django

Структуру файлів фреймворку Flask наведено на рисунку 2.

```
.
├── README.md
├── app
│   ├── __init__.py
│   ├── controllers
│   │   ├── __init__.py
│   │   └── users_controller.py
│   ├── jobs
│   │   └── user_job.py
│   ├── models
│   │   └── user.py
│   └── views
│       └── users
│           └── index.html
├── bin
│   └── start.sh
├── celery_worker.py
├── instance
│   └── development.cfg
└── main.py
```

Рис. 2. Структура файлів фреймворку Flask

На час завантаження видимих елементів програми впливає продуктивність фреймворку для

створення програми разом із розміром створених файлів шаблонів HTML. Навіть якщо розмір шаблону невеликий, завантаження окремих елементів може сповільнитися через низьку продуктивність фреймворку, який використовується для створення веб-програми.

Виконані вимірювання складаються з двох тестів. Обидва з них тестували веб-додатки під великим навантаженням. Перший тест вимірював час доступу до певного шаблону під навантаженням, що виникає внаслідок численних запитів на повторний доступ.

Таблиця 1

Час, витрачений на розробку програми

	Час написання, хв		**Виправлення помилок, хв**		**Загальний час, хв**	
	Flask	Django	Flask	Django	Flask	Django
Вхід	80	50	0	30	80	80
Реєстрація	105	40	0	0	105	40
Активність	60	40	30	0	90	40
Додавання записів	25	35	5	0	30	35
Видалення записів	25	30	0	20	25	50
Оновлення записів	25	30	0	0	25	30
Профіль користувача	60	30	0	5	60	35
Видалення профілю	15	10	5	0	20	10
Оновлення профілю	60	25	15	0	75	25
Додавання файлів	50	25	5	0	55	25
Видалення файлів	25	20	20	0	45	20
Оновлення файлів	50	10	0	0	50	10
Перевірка даних	10	10	0	0	10	10
Панель адміністратора	30	10	0	5	30	15
Відновлення паролю	10	10	0	0	10	10
Разом	630	375	80	60	710	435

До існуючої програми було внесені чотири зміни: видалення модуля, відповідального за додавання файлів, додавання списку, що по черзі висвітлює оголошення адміністратора, об'єднання шаблону контакту з описом призначення програми та додавання календаря із записами користувача до сервера про отримання доступу до випадкових ресурсів веб-програми. Для проведення вимірювань був використаний інструмент під назвою JMeter.

Для порівняння також використовувався критерій – типова кількість файлів, необхідних для виконання веб-додатку. Цей критерій показує, наскільки складною може бути конфігурація багатофункційної веб-програми. Менша кількість файлів може означати, що розробнику буде легше налаштувати та краще орієнтуватись у вихідному коді програми. Якщо певна функційність програми вимагає, щоб для роботи було налаштовано занадто багато файлів, то наступні зміни займуть більше часу для програміста.

Виконані вимірювання включають кількість файлів, необхідних для реалізації окремих функцій, а також середню кількість файлів, необхідних для реалізації будь-якої з них.

Досить важливо знати, чи розроблена програма відповідає стандартам W3C. Для того, щоб веб-додаток правильно відображався у різних веб-браузерах, необхідно переконатися, що він створений відповідно до загальноприйнятих правил. Відповідність стандартам виконання веб-додатків було перевірено за допомогою засобу перевірки W3C. Отримана кількість помилок разом із середньою кількістю порушень представлена у наступних таблицях. Результати розбиті на підрозділи, які відповідають прийнятим критеріям. Порівняння проводилося з використанням обраних фреймворків для розробки веб-додатків. Були проаналізовані фреймворки Django та Flask.

Таблиця 2

Час модифікації програми

	Необхідний час, хв		Кількість модифікованих файлів	
	Flask	*Django*	*Flask*	*Django*
Видалення модуля	3	5	9	10
Додавання списку	22	30	11	10
Об'єднання шаблону контакту з описом призначення програми	8	15	14	11
Додавання календаря із записами користувача	13	15	10	7
Разом	46	65	44	38

Функційні можливості програми показані в тому порядку, в якому вони відображаються у представлених таблицях. Зростання ефективності помітне через скорочення часу впровадження на наступних етапах розробки веб-додатків.

Час, витрачений на розробку програми, вимірювався на основі однієї функції. Результати поділено на три групи: час розробки, час виправлення помилок та сума цих двох показників. Час вимірювався у хвилинах.

Розмір записаного коду поділений на розмір файлів конфігурації, розмір файлу перегляду та розмір шаблону HTML. Результати представлені в байтах, окремо для кожної з функційних можливостей.

Результати вимірювання для фреймворку Django не показують розмір файлів конфігурації, оскільки у всій програмі доступний лише один файл конфігурації. Натомість представлені розміри файлів, в яких працює механізм перевірки даних.

Таблиця 3

Розмір написаного коду

	Файл конфігурації, байт	Перевірка даних, байт	Файл перегляду, байт		Шаблон HTML, байт		Разом, байт	
	Flask	Django	Flask	Django	Flask	Django	Flask	Django
Вхід	549	591	1121	1427	1246	816	2916	2834
Реєстрація	500	0	577	352	131	553	1208	905
Активність	673	0	1749	2555	2871	1978	5293	4533
Додавання записів	673	1984	2049	2608	3019	1632	5741	6224
Видалення записів	686	0	1550	0	932	1483	3168	1483
Оновлення записів	941	391	1733	3505	896	1130	3570	5026
Профіль користувача	612	0	2036	2304	1570	1564	4218	3868
Видалення профілю	670	0	1452	2304	1090	1760	3212	4064
Оновлення профілю	747	2256	4137	1745	3372	1752	8256	5753
Додавання файлів	684	697	1810	6242	1189	1046	3683	7985
Видалення файлів	602	444	1203	2779	749	911	2554	4134
Оновлення файлів	762	0	2005	1468	1045	1621	3812	3089
Перевірка даних	685	386	1994	3796	780	1129	3459	5311
Панель адміністратора	679	0	2067	2601	1038	1612	3784	4213
Відновлення паролю	680	385	3254	4122	1351	1120	5285	5627
Середнє	676,2	475,6	1915,8	2520,5	1418,6	1340,4	4010,6	4336,6

Загальний аналіз фреймворків Django та Flask дають подібні результати. Слід відзначити невелику перевагу Flask.

Розмір створеного HTML-коду вимірювався в байтах на функцію. Запити сторінок були відправлені на сервер і був виміряний розмір отриманих даних. Також був врахований середній розмір однієї сторінки.

Таблиця 4

Розмір створених файлів

	Flask, байт	*Django*, байт
Вхід	2640	1181
Реєстрація	2821	1353
Активність	8450	4415
Додавання записів	5442	5500
Видалення записів	5147	3407
Оновлення записів	3716	2142
Профіль користувача	3334	2021
Видалення профілю	3886	2664
Оновлення профілю	6004	5955
Додавання файлів	4282	3295
Видалення файлів	3628	2407
Оновлення файлів	3044	1811
Перевірка даних	3562	2075
Панель адміністратора	3027	1801
Відновлення паролю	3931	2061
Середнє	4194,27	2805,87

Результати цього вимірювання надають перевагу фреймворку Django. Фреймворк для побудови програми Flask в отриманих HTML-файлах містить додаткові теги та скрипти. Механізм роботи обробки формулярів також працює по-різному. Flask зберігає на сторінках більше інформації про поля формуляру. Розмір HTML-файлів, створених за допомогою фреймворку Flask, виявляється значно більшим.

Було проведено два тести, щодо ефективності проаналізованих фреймворків розробки веб-додатків. Один з тестів імітував одночасну роботу 50 користувачів, які намагались отримати доступ до однієї сторінки 100 разів (5000 запитів). Другий тест моделював роботу 100 користувачів, які періодично переглядали всі доступні сторінки 10 разів (всього 15000 запитів – 100 користувачів * 15 доступних сторінок функціоналу * 10 циклів). Для першого вимірювання була обрана головна сторінка панелі користувача, оскільки це найбільша доступна сторінка функційності в розробленому додатку.

Тестування продуктивності проводилося за допомогою JMeter, і програми розміщувалися на тому ж веб-сервері Apache. Вимірювання були проведені після 5000 запитів на тестування для усунення потенційних ускладнень у майбутньому.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

У даній роботі:

Виділено сучасні фреймворки на основі мови програмування Python.

На основі аналітичного підходу, визначено критерії оцінки веб-фреймворків, розроблених на основі мови програмування Python. Вибрані критерії дозволяють якісніше та об’єктивніше оцінити фреймворки.

Порівняльний аналіз фреймворків Django та Flask показав, що фреймворк Django є кращим, якщо порівнювати час, витрачений на розробку програм. Аналіз продуктивності фреймворків за об’ємом програмного коду надає перевагу фреймворку Django. Фреймворк для побудови програми Flask в отриманих HTML-файлах містить додаткові теги та скрипти. Механізм роботи обробки формулярів також працює по-різному. Flask зберігає на сторінках більше інформації про поля формуляру. Розмір HTML-файлів, створених за допомогою фреймворку Flask, виявляється значно більшим.

Література

1. Belal M., Khedr A., Gohar A. (2012). Frameworks Between Components and Objects, Avanced Computing, vol. 3, №. 5, pp. 9–17, 2012. https://www.semanticscholar.org/paper/FRAMEWORKS-BETWEEN-COMPONENTS-AND-OBJECTS-Belal-Khedr/79a33b455aea7ddac73881da00ea050b7c36db4e

2. Carzaniga A., Fuggetta A., Hall R. S., Heimbigner D., van der Hoek A., and Wolf A. L. (1998). A Characterization Framework for Software Deployment Technologies, Technical Report Department of Computer Science, Boulder, Colorado, April 1998 pp. 1–6. https://www.semanticscholar.org/paper/A-Characterization-Framework-for-Software-Carzaniga-Fuggetta/738bb20ec7cf51d2ec2ac854349840e741461b2e

3. Miłosierny Bartosz, Dzieńkowski Mariusz (2021). Analiza porównawcza szkieletów do budowy aplikacji internetowych w ekosystemie Node.js, Journal of Computer Sciences Inatitute, vol. 18, s. 42–48. https://doi.org/10.35784/jcsi.2423

4. Andri Sunardia, Suharjitoa (2019). MVC Architecture: A Comparative Study Between Laravel Framework and Slim Framework in Freelancer Project Monitoring System Web Based, Procedia Computer Science, Volume 157, pp. 134–141. https://doi.org/10.1016/j.procs.2019.08.150

5. Syamsul Syafiq, Maslina Daud, Hafizah Hasan, Ahmad Zairi, Shazil Imri, Ezaini Akmar, Norbazilah Rahim (2018). Comparison of Web Development Using Framework over Library, Systems Engineering, vol. 12, No 3, pp. 215–218. https://scholar.archive.org/work/3lfaywaelrasdo4sivuak5yvaa

6. Wodyk R., Skublewska-Paszkowska M. (2020). Performance comparison of relational databases SQL Server, MySQL and PostgreSQL using a web application and the Laravel framework. Journal of Computer Sciences Institute, Vol 17, 2020, pp. 358-364. https://doi.org/10.35784/jcsi.2279

7. Kovalenko V. O. Porivnialnyi analiz naiposhyrenishykh freimvorkiv dlia PHP u sferi veb-rozrobok. Naukovi zapysky: Zb. nauk. pr. Kirovohrad: KNTU, 2010. Vyp. 10, ch. 3. S. 75-78.http://dspace.kntu.kr.ua/jspui/handle/123456789/5594

8. Jibril Adamu, Raseeda Hamzah, Marshima Mohd Rosli (2020). Security issues and framework of electronic medical record: Areview. Bulletin of Electrical Engineering and Informatics Vol. 9, No 2, 2020, pp. 565–572

9. Roman Baida, Maksym Andriienko, Małgorzata Plechawska-Wójcik (2020). Analiza porównawcza wydajności frameworków Angular oraz Vue.js Journal of Computer Sciences Institute, Vol. 14, 2020 ss. 59–64 http://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-d67eb923-19f0-4c95-aa9d-e1b017ada2d8

10. Svensson A. Speed Performance Comparison of JavaScript MVC Frameworks. Blekinge Institute of Technology, 2015, p. 36.

11. Petukhova E. Sitecore JavaScript Services Framework Comparison. Åbo Akademi University, Science and Engineering, 2019 p. 74.

12. Kaluža M., Troskot K., Vukelić B. (2018) Comparison of Front-End Frameworks for Web Applications development. Zbornik Veleučilišta u Rijeci, Vol. 6 No. 1, (2018), pp. 261–282.