УДК 004.4'2

Р. Я. Гелей
І. В. Богач

ОГЛЯД ФРЕЙМВОРКІВ PYTHON ДЛЯ ВЕБ-РОЗРОБКИ, ЇХ СИЛЬНІ ТА СЛАБКІ СТОРОНИ

Вінницький національний технічний університет

Анотація
Розглянуто найбільш популярні фреймворки Python для розробки веб-додатків серед яких Django, FastAPI, Flask, а саме принципи їх роботи, сильні та слабкі сторони.
Ключові слова: Python, Django, FastAPI, Flask, веб-додаток, фреймворк, веб-розробка.

Abstract
The most popular Python frameworks for the development of web applications, including Django, FastAPI, and Flask, are considered, including their principles of operation, strengths, and weaknesses.
Keywords: Python, Django, FastAPI, Flask, web-application, framework, web development.

Вступ

Веб-розробка — це процес створення веб-сайту або веб-додатку. Розробка веб-додатку включає в себе як фронтенд (клієнтську частину), так і бекенд (серверну частину) розробку. Для front-end розробки найпопулярніші мови програмування – це HTML, CSS, JavaScript та TypeScript. Для backend – Python, Java, PHP, Ruby, JavaScript та C#. Веб-розробка на Python користується великою популярністю завдяки великій кількості фреймворків і бібліотек, які спрощують процес створення веб-додатків.

Фреймворк (framework) – це набір інструментів, бібліотек та правил, який використовується для створення програмних додатків. Фреймворк призначений для спрощення розробки додатків, оскільки він надає готові рішення для поширених завдань. Це дозволяє програмістам зосередитися на створенні бізнес-логіки програми, а не витрачати час на написання коду для вирішення спільних завдань. [1] Найпопулярніші фреймворки Python – це Django, FastAPI та Flask.

Кожен фреймворк має свої особливості, сильні та слабкі сторони. Вибір правильного фреймворку для веб-розробки є критичним рішенням, яке впливає на ефективність розробки, продуктивність додатку, масштабованість та підтримку. Для того, щоб правильно обрати фреймворк для розробки веб-додатків потрібно мати розуміння принципів роботи основних з них, що і буде розглянуто далі.

Результати дослідження

Django — високорівневий відкритий Python-фреймворк (програмний каркас) для розробки вебсистем. Архітектура Django подібна на «Модель-Вигляд-Контролер» (MVC). Однак, те що називається «контролером» в класичній моделі MVC, в Django називається «вигляд» (англ. view), а те, що мало б бути «виглядом», називається «шаблон» (англ. template). Таким чином, MVC розробники Django називають MTV («Модель-Шаблон-Вигляд»)[2]. Часто використовується для розробки соціальних мереж, додатків електронної комерції, CMS-систем та веб-додатків, які потрібно швидко розробити.

Переваги Django[4]:

- Швидка розробка. Django дозволяє швидко створювати веб-програми завдяки великій кількості вбудованих інструментів та функцій(готові компоненти для обробки аутентифікації, адмін-панелі, форм, ORM, і багато іншого). Це зменшує кількість коду, який слід писати з нуля.
- Безпека. Django надає безліч вбудованих засобів для забезпечення безпеки, таких як захист від CSRF, XSS, SQL-ін'єкцій та інших поширених уразливостей.
- Масштабованість. Django підходить як для невеликих, так і для великих проектів і може ефективно масштабуватись завдяки своїй модульній архітектурі.

- Спільнота і документація. Django має велику і активну спільноту, а це означає, що є безліч ресурсів, бібліотек, сторонніх пакетів. Велика документація дозволяє розробникам легко розпочати роботу та знайти відповіді на свої запитання.

Недоліки Django:

- Крива навчання. Django може призвести до більш крутої кривої навчання для новачків, ніж мікрофреймворки, такі як Flask.
- Монолітність. Всеосяжний характер Django може спричинити непотрібну складність для простих проектів(наприклад Django рідко підходить як сервіс в мікросервісній архітектурі).
- Структура. Чітка структура Django може не тільки полегшувати розробку веб-додатків, але і навпаки, якщо розробнику потрібно написати щось с самого нуля.
- Швидкість. Менше підходить для додатків, де потрібна висока швидкість обробки великої кількості одночасних запитів, адже Django синхронний фреймворк.

FastAPI – це сучасний фреймворк для створення веб-додатків та API мовою Python. Він поєднує продуктивність і простоту Python з високою продуктивністю асинхронного програмування, що робить його відмінним вибором для створення масштабованих і ефективних веб-сервісів. FastAPI немає чіткої структури, що дає змогу створювати архітектуру з нуля. Підходить для розробки веб-API, мікросервісів, додатків, які працюють в реальному часі або які обробляють безліч одночасних запитів.

Переваги FastAPI:

- Висока продуктивність. FastAPI відомий своєю винятковою продуктивністю, асинхронними можливостями та ефективною обробкою запитів, що робить його придатним для проектів із високими вимогами до продуктивності.
- Автоматичне документування. Автоматичне створення інтерактивної документації API спрощує тестування та використання API, заощаджуючи час та зусилля на ведення окремої документації.
- Безпека типів. Використання підказок типів підвищує безпеку та читабельність коду, спрощуючи виявлення помилок у розробці.
- Зростаюча екосистема. Екосистема розширень та бібліотек FastAPI, хоч і не така зріла, як деякі старі платформи, розширюється, пропонуючи більше можливостей та підтримки.

Недоліки Django:

- Крива навчання: розробники, які погано знайомі з асинхронним програмуванням, можуть зіткнутися з необхідністю навчання при адаптації до асинхронних функцій FastAPI.
- Зрілість: у порівнянні з добре зарекомендованими фреймворками, такими як Flask і Django, FastAPI є відносно новим, що може призвести до періодичних змін та оновлень.
- Менш готових рішень: Існують менш розвинені екосистеми плагінів та розширень, що може вимагати більше зусиль для розробки деяких функцій.

Flask – мікрофреймворк для веб-додатків, створений з використанням Python. Його основу складає інструментарій Werkzeug та рушій шаблонів Jinja2. Flask називається мікрофреймворком, оскільки він не вимагає спеціальних засобів чи бібліотек. У ньому відсутній рівень абстракції для роботи з базою даних, перевірки форм або інші компоненти, які надають широковживані функції за допомогою сторонніх бібліотек. Однак, Flask має підтримку розширень, які надають додаткові властивості таким чином, наче вони були доступні у Flask із самого початку. [3] Flask немає чіткої структури, що дає змогу створювати архітектуру з нуля. Flask добре підходить для створення веб-додатків малого та середнього розміру, де простота та швидка розробка є пріоритетами, також підходить для розробки мікросервісів та веб-API.

Переваги Flask:

- Простота: мінімалістичний дизайн Flask і простий API полегшують його вивчення та розуміння, що робить його відмінним вибором для новачків та розробників, які воліють менш самовпевнене середовище.
- Гнучкість: Flask надає розробникам свободу вибору компонентів та бібліотек, дозволяючи їм адаптувати свої програми до конкретних потреб.
- Мінімальні накладні витрати: Flask має невелику кодову базу та мінімальні накладні витрати, що забезпечує високу продуктивність та ефективне використання ресурсів.
- Підтримка спільноти. Flask має активну спільноту, яка пропонує навчальні посібники, документацію та підтримку для розробників.

Недоліки Flask:

- Обмежені вбудовані функції: Flask надає тільки найнеобхідніше, а це означає, що розробникам, можливо, доведеться покладатися на сторонні розширення або бібліотеки для більш сучасних функцій.
- Крива навчання для складних проектів. Хоча запуск Flask простий, складніші проекти можуть вимагати додаткового планування та структури, які розробники повинні реалізувати самостійно.
- Масштабованість: Flask не надає вбудованих інструментів для масштабування додатків, що робить його менш придатним для великих проектів

Висновки

При виборі між Django, Flask та FastAPI важливо узгодити свій вибір з унікальними вимогами вашого проекту та досвідом вашої команди. Django підходить для різного розміру додатків, для яких підходить використання готових інструментів Django, а Flask відмінний вибір для простих та невеликих веб-додатків, прототипів та проектів, де важлива гнучкість та мінімалізм. FastAPI чудово справляється з високопродуктивними API та додатками реального часу, особливо для проектів з високими вимогами до паралелізму.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Фреймворк [Електронний ресурс] – Режим доступу до ресурсу: https://brainlab.com.ua/uk/blog-uk/shho-take-frejmvork-poyasnyuyemo-prostymy-slovamy.
2. Django, Wikipedia [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Django.
3. Flask, Wikipedia [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Flask.
4. Django vs Flask vs FastAPI[Електронний ресурс] – Режим доступу до ресурсу: https://www.capitalnumbers.com/blog/django-vs-flask-vs-fastapi/.

Гелей Роман Ярославович *– студент групи ЗАКІТ-20Б, кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: geleyroman941@gmail.com*

Богач Ілона Віталіївна *– к.т.н., доцент кафедри автоматизації та інтелектуальних інформаційних технологій, факультет комп'ютерних систем і автоматики, Вінницький національний технічний університет, м.Вінниця, e-mail: ilona.bogach@gmail.com*

Helei Roman Yaroslavovych *– student of ZAKIT-20B group, Department of Automatization and Intellectual Informational Technologies, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: geleyroman941@gmail.com*

Bogach Ilona Vitaliivna *– Associate Professor of Automation and Intelligent Information Technologies, Faculty of Intelligent Information Technology and Automation Department, Vinnytsia National Technical University, Vinnytsia, e-mail: ilona.bogach@gmail.com*