Відкриті інформаційні та комп'ютерні інтегровані технології, № 107, 2026 ISSN 2071-1077(print)
ISSN 2663-2411(online

doi: 10.32620/oikit.2026.107.0

УДК 004.4

Т. Л. Столяренко

Дослідження еволюції Python-фреймворків та їх впливу на розвиток інформаційних технологій

Харківський національний економічний університет імені Семена Кузнеця

У статті проаналізовано розвиток екосистеми Python-фреймворків та оцінено їхній вплив на різні напрями інформаційних технологій. Досліджено перехід від ранніх протоколів CGI до сучасних асинхронних архітектур на базі ASGI, що забезпечують високу продуктивність у високонавантажених системах. Виявлено закономірності впливу архітектурних рішень (монолітних та мікросервісних) на стабільність, масштабованість та безпеку інформаційних систем. У фокусі – категорії веб-фреймворків (Django, Flask, FastAPI тощо), фреймворків для машинного та глибокого навчання (TensorFlow, PyTorch), засобів автоматизації тестування (Selenium, Robot Framework) і DevOps-інструментів (Ansible, Fabric), а також інструментарій для створення API та графічних інтерфейсів. Детально висвітлено ключові етапи еволюції цих фреймворків, напрями їхнього застосування (розробка ПЗ, наука про дані, хмарні обчислення, автоматизація, кібербезпека, системна інтеграція) та приклади впливу на IT-галузь. Згідно з опитуваннями спільноти Python за 2024–2025 рр., лідерами серед веб-фреймворків є FastAPI (38 %), Django (35 %) і Flask (34 %), а серед бібліотек ML – scikit-learn (68 %), PyTorch (66 %) та TensorFlow (49 %). Наприклад, Django (створений у 2003–2005 рр.) спроєктовано для швидкого створення складних базоорієнтованих сайтів, а Flask (перший реліз 2010 р.) став надпопулярним мікрофреймворком (використовується у Pinterest, LinkedIn тощо). TensorFlow (2015) і PyTorch (2016) забезпечили шалену експансію засобів штучного інтелекту. У сфері автоматизації Selenium (від 2004) і Robot Framework (від 2005) стали де-факто стандартами тестування веб- та системних інтерфейсів, а DevOps-інструменти Ansible (2012) та Fabric суттєво спростили конфігурування систем і розгортання ПЗ. З іншого боку, Python-фреймворки для наукових обчислень (пакети Pandas, NumPy, SciPy) й ML-завдань надали ключові інструменти науки про дані. Загалом, вивчено сучасні тенденції та практичні приклади застосування Python-фреймворків у IT, що демонструють їхню важливість у прискоренні розробки та впровадженні інновацій [1-6,10].

Ключові слова: Python, фреймворки, еволюція API, WSGI, ASGI, мікросервісна архітектура, Django, Flask, TensorFlow, PyTorch, Selenium, Robot Framework, Ansible, Fabric, розробка програмного забезпечення, наука про дані, хмарні обчислення, автоматизація, кібербезпека.

Вступ

Сучасна ера цифрової трансформації висуває безпрецедентні вимоги до інженерії програмного забезпечення. Швидкість обробки даних, здатність систем до горизонтального масштабування та мінімізація затримок у реальному часі стали критичними факторами успіху для будь-якого технологічного продукту. У цьому контексті мова програмування Python пройшла шлях від допоміжного інструменту автоматизації до центрального елемента стеку розробки найбільш складних інформаційних систем. Проте, попри внутрішню елегантність мови, її реальна потужність у веб-розробці та системній інтеграції реалізується через екосистему фреймворків, які пройшли тривалий та складний шлях еволюції.

1. Постановка проблеми

Проблема дослідження полягає у необхідності систематизації знань про

еволюційний розвиток цих інструментів, оскільки вибір фреймворка сьогодні не є лише питанням зручності програміста, а стратегічним архітектурним рішенням, що визначає життєздатність системи на роки вперед. Швидка зміна парадигм — від синхронних блокуючих запитів (WSGI) до асинхронних неблокуючих систем (ASGI) — створила розрив у розумінні оптимальних підходів до проектування систем у таких критичних доменах, як фінансова аналітика, охорона здоров'я та кібербезпека.

Більше того, динамічність екосистеми Python несе в собі приховані ризики: швидка еволюція API часто призводить до проблем сумісності, що генерує значний технічний борг у довгостроковій перспективі. Відтак, комплексне дослідження еволюції фреймворків та їхнього впливу на IT-галузь є актуальним як з теоретичної точки зору (розуміння архітектурних трендів), так і з практичної (оптимізація витрат на розробку та підтримку).

Python – одна з провідних мов програмування в IT (за даними опитувань, близько половини розробників використовують Python в роботі). Її популярність обумовлена зрозумілим синтаксисом і потужною екосистемою бібліотек і фреймворків, яка охоплює практично всі сфери сучасних інформаційних технологій. Проте у наукових публікаціях бракує комплексного огляду історії розвитку Python-фреймворків та їхнього впливу на різні галузі. Саме тому актуальним є завдання систематизації ключових етапів еволюції цих інструментів та оцінка їхнього впливу на наукові і практичні задачі IT, зокрема в розробці ПЗ, науці про дані, хмарних обчисленнях, автоматизації, кібербезпеці та системній інтеграції [7,12].

2. Аналіз останніх досліджень та публікацій

Питання розвитку Python та його інфраструктури розглядалися багатьма дослідниками як в академічному середовищі, так і в межах великих технологічних корпорацій. Фундаментальні дослідження Чжана та співавторів фокусуються на еволюції API та проблемах сумісності, виявляючи, що Python-фреймворки змінюються значно інтенсивніше, ніж їхні аналоги в Java або C#. Це створює унікальний ландшафт, де гнучкість мови поєднується з високим темпом інновацій, але водночас вимагає від розробників постійного моніторингу змін [22, 48].

Важливий сегмент досліджень присвячений порівняльному аналізу продуктивності. Роботи 2023–2025 років надають емпіричні докази переваг FastAPI над традиційними фреймворками у сценаріях з високою щільністю запитів. Водночас, академічні публікації щодо Django підкреслюють його незамінність у створенні надійних корпоративних платформ завдяки вбудованим механізмам безпеки та комплексному управлінню базами даних [21, 23, 49].

В українському науковому просторі питання застосування Python-технологій досліджуються у контексті спеціалізованих систем. Наприклад, використання мови у геоінформаційних системах для муніципального управління та моделювання сільськогосподарських процесів демонструє високу адаптивність фреймворків до специфічних галузевих потреб. Дослідження кібербезпеки в Україні також вказують на Python як на ключовий інструмент для створення додатків захисту інформації та аналізу мережевих загроз. Проте, попри наявність окремих досліджень, цілісний аналіз еволюції фреймворків як рушійної сили розвитку IT-галузі залишається фрагментарним [26, 27, 40, 46, 47].

Незважаючи на розповсюдженість Python, у науковій літературі спостерігається дефіцит узагальнених праць про Python-фреймворки як кластер

інструментів. Існують маркетингові та інформаційні огляди, що висвітлюють популярність окремих технологій, але вони не охоплюють весь спектр фреймворків. Наприклад, останні опитування спільноти Python (JetBrains/Python Software Foundation 2024–2025) вказують на домінування FastAPI (38 %), Django (35 %) та Flask (34 %) у веб-розробці, а серед ML-фреймворків – scikit-learn (68 %), PyTorch (66 %) і TensorFlow (49 %). У сфері ETL-інструментів відзначено швидке зростання завантажень Apache Airflow і загальну домінацію Python у конструюванні дата-пайплайнів. Також виявлено, що 77 % спеціалістів з аналізу даних використовують бібліотеку Pandas. Тенденції останніх років (зокрема, збільшення ролі штучного інтелекту в опитуваннях) підтверджують, що фреймворки для ML/AI стають центральними інструментами наукових досліджень. У статті враховано всі ці публікації та статистичні дані, однак обрано мету надати глибокий науковий огляд еволюції та ролі саме фреймворків Python (не лише окремих бібліотек) в IT [10, 11].

Метою дослідження є виявлення та систематизація ключових етапів еволюції Python-фреймворків та оцінка їхнього впливу на розвиток різних напрямів інформаційних технологій, аналіз переходу від синхронних до асинхронних архітектур та оцінка їхнього впливу на сучасні парадигми розробки інформаційних технологій, включаючи мікросервіси, штучний інтелект та високонавантажені хмарні системи. Задачами дослідження є: 1) описати хронологічні етапи розвитку ключових категорій фреймворків (веб, ML/DL, автоматизація, DevOps, API, GUI тощо); 2) виявити суттєві зміни у їхній популярності та архітектурі; 3) оцінити, як поява та вдосконалення цих фреймворків вплинули на практики розробки ПЗ, науки про дані, хмарних сервісів, автоматизації і безпеки; 4) намітити перспективні напрями подальших досліджень у контексті зростання екосистеми Python.

3. Дослідження еволюції Python-фреймворків та їх впливу на розвиток інформаційних технологій

3.1. Генезис та перші кроки: від CGI до перших фреймворків

Еволюція веб-технологій на базі Python почалася з епохи CGI (Common Gateway Interface). У середині 1990-х років це був стандартний спосіб взаємодії веб-сервера з виконуваними скриптами. Механізм роботи CGI передбачав запуск нового процесу інтерпретатора Python для кожного вхідного HTTP-запиту. Це створювало величезні накладні витрати на рівні операційної системи. Якщо система отримувала 1000 запитів за секунду, вона мала ініціалізувати 1000 процесів, що призводило до вичерпання ресурсів RAM та CPU за лічені миті.

Математично ефективність CGI можна виразити через затримку системи T_{total}:

$$T_{total} = T_{spawn} + T_{logic} + T_{cleanup}$$

де T_{spawn} — час на ініціалізацію процесу Python, який часто перевищував час виконання самої логіки T_{logic}.

Розуміння цієї неефективності призвело до появи FastCGI та mod_python для Apache. Ці технології дозволяли утримувати процеси в пам'яті (persistence), що значно скорочувало час відповіді. Саме в цей період з’явився Zope (1999) — перший «динозавр» серед Python-фреймворків. Він запропонував концепцію об’єктної публікації, де кожен URL відповідав об’єкту в ієрархічній базі

даних ZODB. Zope був занадто складним та монолітним, але він заклав фундамент для розуміння того, що веб-додаток потребує структурованої архітектури [17, 29, 30].

3.2. Стандартизація через WSGI: Епоха Django та Flask

Справжнім проривом у розвитку інформаційних технологій на Python стало впровадження стандарту WSGI (Web Server Gateway Interface) у 2003 році (PEP 333). WSGI створив універсальний «контракт» між веб-сервером (наприклад, Gunicorn або uWSGI) та веб-фреймворком. Це дозволило розробникам змінювати сервери або фреймворки без переписування всієї логіки додатка [1,2,24,25,33].

На цьому фундаменті виникли два титани розробки:

1. Django (2005): Створений у редакції газети, він втілив філософію «batteries-included» (все включено). Django автоматизував створення адмін-панелей, роботу з базами даних (ORM) та безпеку. Це дозволило бізнесу створювати складні системи у рекордно короткі терміни.

2. Flask (2010): Виник як жарт на перше квітня, але став найпопулярнішим мікрофреймворком. Його філософія — мінімалізм. Flask не нав’язує структуру, дозволяючи розробнику обирати власні інструменти для БД, авторизації та шаблонів.

Таблиця 1

Порівняння традиційних синхронних фреймворків

Критерій порівняння	Django (WSGI)	Flask (WSGI)
Архітектурний підхід	Монолітний, структурований	Модульний, вільний
Рівень абстракції БД	Високий (вбудована ORM)	Низький (через SQLAlchemy)
Вбудована безпека	CSRF, SQLi, XSS захист «з коробки»	Потребує сторонніх плагінів
Масштабованість	Добра для великих монолітів	Чудова для мікросервісів
Навчання	Потребує тривалого часу	Дуже низький поріг входу
Критерій порівняння	Django (WSGI)	Flask (WSGI)

Ця епоха характеризувалася синхронним виконанням коду. Кожен запит займав один потік (thread). Це було прийнятно для традиційних веб-сайтів, але з появою Web 2.0 та Real-time застосунків (чати, ігри, стрімінг) виникла потреба в асинхронності.

Веб-фреймворки Python Фреймворки для веб-розробки – одна з найстаріших і найбільших категорій Python-інструментів. Уже на початку 2000-х з’явились Tornado (Facebook/2009), web2py (2007), Pyramid (2005), TurboGears (2005) тощо. Проте найвпливовішим став Django, розроблений у команді A. Holovaty і S. Willison. Django було створено восени 2003 року, а перший публічний реліз під BSD-ліцензією відбувся в липні 2005 року. Фреймворк спроєктовано для спрощення створення складних, базоорієнтованих веб-додатків: він забезпечує архітектуру Model-Template-View, ORM, систему шаблонів, механізм кешування та інтернаціоналізації. Основні принципи Django –

ISSN 2071-1077(print)
ISSN 2663-2411(online

повторне використання компонентів, мінімізація «бойлерплейт»-коду та швидка розробка (принцип DRY). Цей підхід зробив Django популярним для створення масштабних вебсервісів: його використовують такі відомі проєкти, як Instagram, Mozilla, Disqus, Bitbucket та інші [1,2,24,25,33], що свідчить про значний вплив Django на розробку веб-ПЗ.

Другим за популярністю є Flask – легкий «мікрофреймворк», створений A. Ronacher групою Pocoo. Перший реліз Flask відбувся 1 квітня 2010 року. У Flask немає власної ORM чи системи шаблонів, але він спрощує розробку через підтримку розширень. Завдяки простоті та невеликому базовому коду Flask набув широкого використання для створення API та невеликих сервісів (його застосовують у таких проєктах, як Pinterest та LinkedIn). За останні роки Flask залишався на піку популярності, щороку отримуючи п’ять-шість тисяч зірочок на GitHub і посідаючи лідируючі позиції в опитуваннях розробників.

Останніми роками відбувається бум асинхронних веб-фреймворків. Зокрема, FastAPI (розробник S. Ramírez, перший реліз – грудень 2018) позиціонується як високопродуктивний фреймворк для побудови API у Python 3.8+. Він автоматично генерує документацію (OpenAPI/Swagger) і широко використовується у мікросервісних та хмарних рішеннях. Опитування Python-спільноти 2025 року показують значне зростання популярності FastAPI – з 29 % у 2023 до 38 % в останньому році, що робить його найпоширенішим веб-фреймворком за останній рік. Водночас Django і Flask втратили частину користувачів (35 % та 34 % відповідно). Це підтверджує тренд на швидкі та масштабовані HTTP-сервіси, особливо в проектах з наукою про дані.

Сьогодні веб-фреймворки надають великий набір готових компонентів для створення сайтів і сервісів. Завдяки цим інструментам програмісти можуть швидше реалізовувати функціональність – наприклад, Django «з коробки» дає модуль адміністратора, а Flask та FastAPI спрощують створення маршрутизованих API. Таким чином, розвиток веб-фреймворків безпосередньо прискорює створення нового ПЗ, знижує кількість помилок (стандартизуючи підхід) та сприяє широкому розповсюдженню інтернет-сервісів (рис. 1) [1,2,24,25,33].

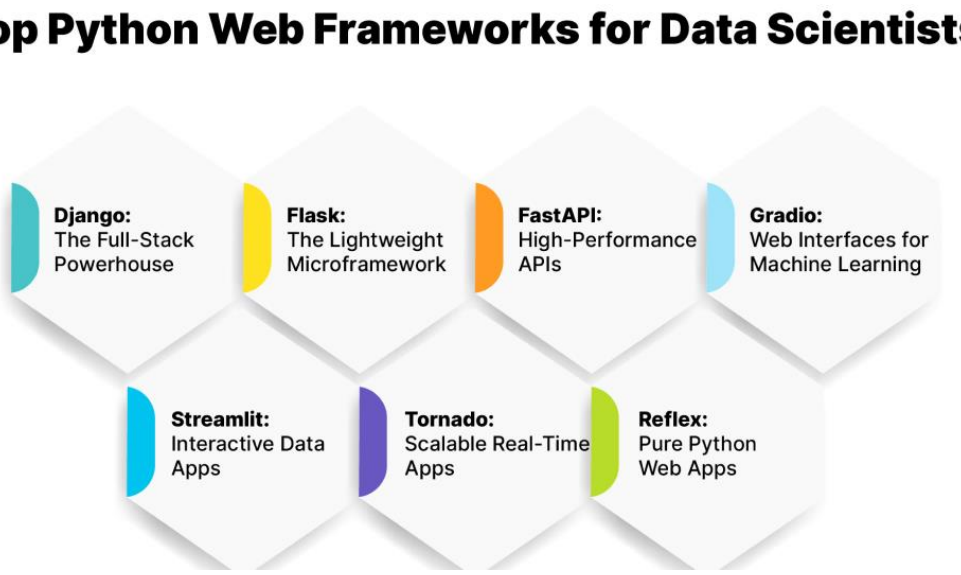

Рис. 1. Top Python Web Fremeworks for Data Scientists

3.3. Асинхронна революція: ASGI та FastAPI

Поява ключових слів `async` та `await` у Python 3.5 ознаменувала початок нової ери. Стандарт WSGI не міг підтримувати асинхронність, оскільки він за своєю суттю є блокуючим. Це призвело до розробки ASGI (Asynchronous Server Gateway Interface). ASGI дозволяє обробляти тисячі конкурентних з'єднань через один потік, використовуючи подієвий цикл (Event Loop). []

Найяскравішим представником цієї хвилі став FastAPI (2018). Він не просто додав асинхронність, а інтегрував сучасні типи Python та валідацію через Pydantic. Вплив FastAPI на IT-галузь був миттєвим: [3,21,34,35,36,37]

- Автоматична документація: Завдяки OpenAPI розробники отримують інтерактивну документацію Swagger без додаткових зусиль.
- Продуктивність: У тестах на пропускну здатність FastAPI наближається до Node.js та Go.
- Типізація: Використання Type Hints дозволяє виявляти помилки на етапі написання коду, що критично для великих команд.

Таблиця 2

Бенчмаркінг продуктивності фреймворків (2025)

Метрика продуктивності	Django (DRF)	Flask	FastAPI (Uvicorn)
Пропускна здатність (req/sec)	~1,551	~2,194	~4,783
Час відповіді (latency, ms)	13.4	5.0	2.5
Споживання RAM на воркер	~40MB	~30MB	~20MB

Такі показники зробили FastAPI де-факто стандартом для розробки мікросервісів та високопродуктивних API у великих корпораціях, таких як Microsoft, Uber та Netflix [3,21,34,35,36,37].

3.4. Вплив на архітектуру мікросервісів та хмарні технології

Еволюція Python-фреймворків безпосередньо стимулювала перехід від великих монолітів до мікросервісної архітектури. Легкість таких інструментів, як Flask та FastAPI, дозволяє створювати малі, незалежні сервіси, які легко упаковуються в Docker-контейнери та розгортаються в Kubernetes.[19,38,39,40,41,42,]

Основні переваги, які Python надав мікросервісам:

1. Гнучкість масштабування: Окремі критичні сервіси (наприклад, обробка платежів) можна масштабувати незалежно від решти системи.

2. Асинхронна комунікація: Використання асинхронних клієнтів дозволяє мікросервісам взаємодіяти без тривалого очікування відповіді, що підвищує загальну відмовостійкість системи.

3. Екосистема хмарних інструментів: Повна сумісність із платформами AWS, Google Cloud та Azure забезпечує швидке впровадження CI/CD пайплайнів.

В українському сегменті IT-розробки мікросервіси на Python є стандартом для аутсорсингових та продуктових компаній. Українські інженери використовують ці технології для створення AI-керованих веб-додатків та блокчейн-платформ.

3.5. Python у штучному інтелекті та Data Science

Неможливо розглядати еволюцію фреймворків у відриві від вибухового

росту штучного інтелекту (AI). Python став «мовою номер один» для AI не лише через свій синтаксис, а й завдяки безшовній інтеграції веб-фреймворків із бібліотеками машинного навчання [4, 5, 16, 32, 43, 44, 45].

Еволюція тут відбувалася у напрямку створення «мостиків» між моделлю та користувачем:

- TensorFlow та PyTorch: Фреймворки для навчання нейронних мереж.
- Scikit-learn: Класичне машинне навчання.
- FastAPI / Flask: Використовуються як сервісна оболонка для інференсу (inference) моделей. Завдяки асинхронності FastAPI може обробляти безліч запитів на передбачення одночасно.

Дослідження вказують, що понад 38% розробників переходять на FastAPI саме через його ефективність у AI-проектах. Більше того, з'явилися вузькоспеціалізовані фреймворки, такі як Gradio та Streamlit, які дозволяють дата-сайентистам створювати веб-інтерфейси для своїх моделей за лічені хвилини без знання JavaScript.

Фреймворки для машинного та глибокого навчання

Сфера науки про дані і штучного інтелекту значно виграла від появи потужних фреймворків ML/DL. Ранні бібліотеки (Theano, 2007; Caffe, 2013) дали старт, але справжній прорив пов'язаний із TensorFlow і PyTorch. TensorFlow (розроблений командою Google Brain) вперше випущено у листопаді 2015 року як відкритий фреймворк для навчання нейронних мереж. Через кілька років (вересень 2019) з'явилася версія TensorFlow 2.0 з новим API, що додало простоти у побудові моделей. PyTorch (розроблений Facebook AI Research) вийшов у вересні 2016 року і швидко завоював популярність завдяки «пайтонічному» інтерфейсу і гнучкій системі автограду. До 2025 року обидва фреймворки залишаються найпоширенішими інструментами глибокого навчання. Зокрема, згідно з опитуваннями, 66 % розробників ML використовують PyTorch, а 49 % – TensorFlow. PyTorch активно розвивається: версія 2.0 (березень 2023) значно пришвидшила виконання коду [4, 5, 16].

Ці фреймворки змінили практики у багатьох галузях: автоматично розпізнавання зображень, обробка природної мови, прогнозування у фінансах тощо. Їх поширення призвело до того, що Python став фактично стандартом для ML/AI-досліджень. Крім TensorFlow і PyTorch, до екосистеми увійшли зручні бібліотеки високого рівня (наприклад, Keras, у складі TensorFlow) та інструменти для роботи з даними (pandas, NumPy, SciPy). За рахунок цього наука про дані стала доступнішою: аналітики і дослідники можуть швидко прототипувати рішення без складних мов програмування.

3.6. Інструменти автоматизації та тестування

Python-фреймворки для автоматизації дедалі частіше використовуються для підвищення якості та безпеки розробки. Особливо це стосується автоматизованого тестування інтерфейсів та робочих процесів. Selenium – один із найвідоміших інструментів для веб-тестування. Історія Selenium почалась у 2004 році (ThoughtWorks, J. Huggins) з інструменту JavaScriptTestRunner для тестування веб-додатків. З часом Selenium перетворився на набір бібліотек, які дозволяють керувати браузерами з різних мов програмування, включаючи Python (через binding'и). Selenium фактично встановив стандарт автоматичного UI-тестування, адже підтримує всі популярні браузери та дає гнучкі можливості для запису/відтворення сценаріїв.

Robot Framework – ще один знаковий фреймворк для автоматизації (keyword-driven testing), створений P. Klärck. Робота над ним почалась у 2004–2005 роках, а у 2008 році він був випущений як відкритий проєкт. Robot Framework надає просту синтаксичну оболонку (використання ключових слів) для написання приймальних тестів різного типу. Його активно використовують як у тестуванні програм (GUI, API), так і для автоматизації системних сценаріїв. Завдяки бібліотекам на Python, Robot легко розширюється і інтегрується з інструментами DevOps.

Крім того, існують численні фреймворки для тестування коду: pytest (2004–2011) забезпечує можливості юніт- та інтеграційного тестування з мінімальним написанням шаблонного коду, а unittest є вбудованим модулем Python. Поєднання тестових фреймворків із CI/CD-системами (Jenkins, GitHub Actions тощо) і DevOps-інструментами (Selenium Grid) суттєво підвищує рівень автоматизації розробки. В цілому, наявність таких фреймворків дозволяє командами більше фокусуватися на коректності та надійності ПЗ, оскільки рутинне тестування стає простішим [6, 7].

3.7. DevOps- та інфраструктурні фреймворки

Розвиток практик DevOps і Infrastructure as Code тісно пов’язаний з появою спеціалізованих Python-інструментів. Однією з найбільш відомих систем є Ansible – платформа автоматизації конфігурації і розгортання. Ansible створено Майклом ДеХаном і вперше випущено в лютому 2012 року. Згідно з описом Red Hat, Ansible – це CLI-утиліта для автоматизації IT, написана на Python: вона дозволяє конфігурувати системи, розгортати програми та оркеструвати складні робочі процеси. Ключова особливість Ansible – відсутність агентів: він підключається до серверів через SSH (за замовчуванням) і виконує модулі на віддалених хостах. Ansible здобув широке поширення в IT-інфраструктурі, оскільки спрощує налаштування середовищ і управління множинними серверами.

Іншим популярним інструментом є Fabric – бібліотека для високорівневого віддаленого виконання команд SSH. Фабрика надає програмний інтерфейс на Python для запуску shell-команд на віддалених хостах і передачі об’єктів між машинами. Як зазначено в офіційній документації, Fabric – це «бібліотека високого рівня Python, призначена для виконання shell-команд віддалено через SSH, повертаючи при цьому зручні Python-об’єкти». Fabric часто використовують для розгортання додатків, виконання скриптів адміністрування, запуску процедур міграції та бекапу.

Крім цих, в Python-спільноті є й інші DevOps-інструменти: SaltStack (2011) – система управління конфігурацією, теж на Python, а також надаються клієнти Boto3 для роботи з AWS, Google Cloud SDK на Python тощо. Загалом, Python-фреймворки в DevOps дають гнучкість в інтеграції і автоматизації. Вони тісно пов’язані з практиками хмарної інфраструктури (автоматичні скрипти, конфігурування контейнерів, CI/CD). Наявність таких фреймворків прискорює цикл розробки та розгортання ПЗ, адже дозволяє об’єднати процеси налаштування середовища з процесом безпосередньої розробки [8, 9, 15].

3.8. Фреймворки для створення API та мікросервісів

Оскільки мікросервісна архітектура і RESTful-сервіси стали стандартом у розробці, з’явились спеціалізовані інструменти для створення API. З уже згаданих

фреймворків варто відзначити FastAPI (2018), який спочатку задумані під API і надає вбудовану валідацію даних (завдяки Pydantic і підказкам типів Python). Також існують рішення на базі Django: наприклад, Django REST Framework (2011) – стороння бібліотека для швидкої побудови REST API на основі Django. Для GraphQL-сервісів розроблено такі проєкти, як Graphene (2017). Хоча для цих фреймворків не завжди потрібен окремий опис окремого, їх розвиток тісно пов’язаний із загальною еволюцією веб-інструментів. Зокрема, FastAPI, завдяки своїй асинхронній природі та перформансу, дуже швидко завоював популярність у проєктах з високими вимогами до швидкості відповіді [23, 49].

3.9. Фреймворки для графічного інтерфейсу користувача (GUI)

Python також має потужні фреймворки для створення GUI-додатків. Найстаршим є Tkinter – стандартний інтерфейс до бібліотеки Tcl/Tk, доступний у складі Python з ранніх версій. Як зазначено в офіційній документації, пакет tkinter є «стандартним інтерфейсом Python до GUI-інструментарію Tcl/Tk». За допомогою Tkinter можна створювати прості віконні програми на різних платформах, хоча його можливості обмежені сучасними вимогами до дизайну.

Більш сучасним кросплатформним фреймворком є Kivy, перший реліз якого вийшов 1 лютого 2011 року орієнтований на створення мультитач-додатків і підтримує запуск на Android, iOS, Linux, Windows і macOS. Він надає власний набір віджетів і використовує OpenGL ES2 для рендерингу графіки. Зазвичай Kivy застосовують в мобільних проєктах або інтерактивних інтерфейсах. Іншими прикладами GUI-фреймворків є PyQt/PySide (Qt for Python, з’явився близько 2009 року) та wxPython (на Qt/C++), які дають можливість створювати багатофункціональні настільні програми[13].

3.10. Вплив на різні галузі інформаційних технологій

Розробка ПЗ. Поява фреймворків суттєво скоротила час розробки ПЗ. Структуровані шаблони архітектури (наприклад, MVC/MTV у Django) і готові компоненти (ORM, системи аутентифікації, кешування) означають, що розробникам доводиться писати менше повторюваного коду. Згідно з описом Django, його ціль – полегшити створення «складних, базоорієнтованих сайтів», роблячи акцент на перерозподілі коду та високій продуктивності розробки. Аналогічно, FastAPI і Flask знижують поріг входження, спрощуючи старт проєкту з невеликою кількістю файлів.

Наука про дані та ML. Завдяки таким фреймворкам, як TensorFlow і PyTorch, Python перетворився на базову мову для DS/AI. Опитування показують, що понад 80 % фахівців у Data Science використовують Python і пов’язані з ним інструменти. Фреймворки машинного навчання дозволяють легко розгортати експериментальні моделі і робити їх масштабованими у виробництві. Наприклад, компанії обробляють величезні масиви даних і використовують ML для прогнозування, а фреймворки забезпечують інструментарій для паралельного навчання на GPU/TPU. Крім того, бібліотеки типу Pandas (77 % використання серед дата-сайєнтистів) спростили підготовку і аналіз даних. Складні ML-програми, що раніше були доступні лише вузькому колу фахівців, тепер створюються швидко і доступно завдяки Python-екосистемі.

Хмарні обчислення та розподілені системи. Python-фреймворки активно застосовуються в хмарних середовищах. Наприклад, Ansible широко

використовується для управління хмарними інстансами (AWS, Azure, GCP) через модулі, що звертаються до відповідних API Apache Airflow (Python-проєкт) став еталоном для оркестрації ETL-пайплайнів у хмарі, про що свідчить його щомісячний мільйонний трафік завантажень. Контейнеризація (Docker) і оркестрація (Kubernetes) також інтегровані з Python (напр., бібліотека docker-py), що робить Python-фреймворки невід'ємною частиною cloud-native підходу. Усі ці технології спрощують масштабування і розгортання програм у хмарі, роблячи Python-фреймворки центральним компонентом у розробці DevOps-процесів.

Автоматизація та роботизація. Фреймворки як Selenium і Robot Framework безпосередньо впливають на автоматизацію тестування, а інструменти як Ansible і Fabric – на автоматизацію розгортання і управління. Це знижує людський фактор у рутинних задачах, підвищуючи швидкість розробки та надійність систем. Крім того, Python-бібліотеки (Paramiko, Netmiko, Scapy та ін.) використовуються у кібербезпеці для сканування, аналізу мережі, автоматизованого виявлення вразливостей. Як відзначає ESET, Python відомий своїми «шпильками» швидкого створення скриптів для сканування портів, аналізу шкідливого ПЗ, управління мережевими пристроями тощо. Можливість швидко писати крос-платформні скрипти і великий набір безкоштовних бібліотек роблять Python-фреймворки ефективним інструментом і в кібербезпеці.

Системна інтеграція. Для інтеграції складних систем часто використовують асинхронні та розподілені фреймворки Python. Наприклад, фреймворк Celery (від 2009 року) – система обробки асинхронних задач за допомогою черг повідомлень – дозволяє розподіляти навантаження між серверами. Celery застосовується у таких масштабних системах, як Instagram, для одночасного опрацювання мільйонів задач. Використання Celery і подібних інструментів (RabbitMQ, Redis, Kafka) показує, що Python-фреймворки активно інтегруються у backend-архітектури, підтримуючи обмін даними між модулями, чергування подій та масштабованість.

Внесок українських розробників та науковців. Україна робить значний внесок у розвиток та застосування Python-технологій. Зокрема, українські вчені інтегрують Python із передовими хмарними сервісами для вирішення глобальних задач.

- Агросектор: У Тернопільському національному педагогічному університеті та інших закладах розробляються моделі кластеризації для сільськогосподарських підприємств, що працюють на базі Python-інфраструктури.
- Екологія та ГІС: Використання Python у поєднанні з Google Earth Engine для реального часу моніторингу стану посівів та лісів.
- Медицина: Застосування Python для аналізу гігапіксельних зображень у діагностиці меланоми.

Ці приклади демонструють, що еволюція фреймворків має прямий вплив на реальний сектор економіки України, підвищуючи ефективність управління та точність прогнозів [10, 11, 12, 20, 26, 27, 28, 46, 47].

3.11. Еволюція API та проблеми підтримки (Maintenance)

Важливим аспектом дослідження є аналіз того, як зміни в коді самих фреймворків впливають на кінцеві продукти. Згідно з дослідженням Чжана, у Python-фреймворках спостерігається висока частка деструктивних змін (breaking changes). 47.3% змін в API пов'язані з видаленням функціоналу, що часто призводить до «падіння» систем після простого оновлення бібліотеки [48].

На відміну від статичних мов, де помилка виявляється компілятором, у Python-проектах ці проблеми проявляються як `AttributeError` або `TypeError` вже у продуктивному середовищі. Для вирішення цієї проблеми були розроблені інструменти на кшталт PYCOMPAT, які використовують статичний аналіз для виявлення потенційних конфліктів сумісності. Це підкреслює зрілість екосистеми: від хаотичного розвитку до системного управління життєвим циклом ПЗ [14, 18, 22, 31, 48]

4. Висновок

Проведене дослідження демонструє, що Python-фреймворки відіграють критичну роль у розвитку інформаційних технологій. Від появи перших веб-фреймворків на початку 2000-х до сучасних ML-бібліотек і інструментів автоматизації – спільною рисою є спрощення і прискорення розробки складних систем. Веб-фреймворки (Django, Flask, FastAPI) зробили створення інтернет-сервісів доступним і стандартизованим, ML-фреймворки (TensorFlow, PyTorch) – поширили AI у дослідженнях і промисловості, а DevOps/тест-фреймворки (Ansible, Selenium, Robot) – автоматизували рутини і покращили якість продуктів. Економічний та соціальний вплив помітний: за останнє десятиліття Python став одним з головних драйверів інновацій у програмуванні та обчисленнях.

Дослідження еволюції Python-фреймворків дозволяє зробити висновок про фундаментальну трансформацію мови з інструменту сценаріїв у потужну платформу для побудови корпоративних та наукових систем. Перехід від синхронних (WSGI) до асинхронних (ASGI) архітектур став відповіддю на виклики сучасної IT-індустрії, що потребує високої конкурентності та низьких затримок.

Ключові висновки:

1. Архітектурна диференціація: На ринку встановився паритет між монолітними рішеннями (Django), що забезпечують безпеку та швидкість розробки стандартних систем, та асинхронними мікрофреймворками (FastAPI), що є ідеальними для мікросервісів та AI.

2. Продуктивність: Асинхронність дозволила Python подолати історичне обмеження продуктивності, зробивши його конкурентним у високонавантажених доменах.

3. Стандартизація: Впровадження WSGI та ASGI стало критичним моментом, що забезпечив сумісність та стимулював ріст екосистеми бібліотек.

4. Вплив на інновації: Python-фреймворки стали головним драйвером демократизації AI та машинного навчання, дозволяючи швидко перетворювати наукові моделі на комерційні продукти.

Перспективи розвитку включають подальшу інтеграцію з технологіями Web 3.0, розвиток безсерверних (serverless) архітектур на базі Python та впровадження інструментів на основі штучного інтелекту для автоматичного написання та оптимізації коду самих фреймворків. Також очікується посилення уваги до безпеки API та автоматизованого управління сумісністю бібліотек у великих проектах. У подальших роботах доцільно відстежувати появу нових Python-фреймворків і технологічних трендів (наприклад, розвиток AI/ML, IoT, blockchain, квантових бібліотек), а також вивчати методики оцінки їхньої ефективності у розробці. Важливим є також аналіз проблем сумісності версій (перехід на Python 3.x, підтримка старого коду) і безпекових ризиків, пов’язаних із загальною екосистемою пакунків (як уразливості PyPI). Таким чином, Python-фреймворки залишатимуться активною областю досліджень, а їхнє постійне

вдосконалення сприятиме розвитку інформаційних технологій у всіх ключових доменах.

Список літератури

1. Django (web framework) [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Django_(web_framework) (дата звернення: 25.12.2025).
2. Flask (web framework) [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Flask_(web_framework) (25.12.2025).
3. FastAPI [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/FastAPI (дата звернення: 25.12.2025).
4. TensorFlow [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/TensorFlow (25.12.2025).
5. PyTorch [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/PyTorch (25.12.2025).
6. History of Selenium [Електронний ресурс] // selenium.dev. – Режим доступу: https://www.selenium.dev/about/history/ (25.12.2025).
7. Robot Framework – The History of Robot Framework [Електронний ресурс] // blog.efficode.com. – Режим доступу: http://blog.efficode.com/the-history-of-robot-framework-3077efc16419 (25.12.2025).
8. Red Hat. How Ansible works [Електронний ресурс] // redhat.com. – Режим доступу: https://www.redhat.com/en/ansible-collaborative/how-ansible-works (25.12.2025).
9. Ansible (software) [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Ansible_(software) (25.12.2025).
10. Framework Training. What the 8th Python Developer Survey means for you! [Електронний ресурс] – frameworktraining.co.uk (24.09.2025). – Режим доступу: https://www.frameworktraining.co.uk/news-insights/what-eighth-python-developer-survey-means (25.12.2025).
11. Integrate.io. Python ETL Framework Usage Trends — 15 Statistics Shaping Data Pipeline Development in 2026 [Електронний ресурс] – integrate.io (26.09.2025). – Режим доступу: https://www.integrate.io/blog/python-etl-framework-usage-trends/ (25.12.2025).
12. Bravo C.A. Gripped by Python: 5 reasons why Python is popular among cybersecurity professionals [Електронний ресурс] // WeLiveSecurity (ESET) (25.04.2024). – Режим доступу: https://www.welivesecurity.com/en/secure-coding/python-5-reasons-popular-cybersecurity-professionals/ (25.12.2025).
13. Python Software Foundation. tkinter — Python interface to Tcl/Tk [Електронний ресурс] // Python Documentation (3.14.2). – Режим доступу: https://docs.python.org/3/library/tkinter.html (25.12.2025).
14. Celery (software) [Електронний ресурс] // Wikipedia. – Режим доступу: https://en.wikipedia.org/wiki/Celery_(software) (25.12.2025).
15. Fabric. Welcome to Fabric! [Електронний ресурс] // fabfile.org. – Режим доступу: https://www.fabfile.org/ (25.12.2025).
16. Python As The Backbone Of Modern And Emerging Technologies - IJCRT.org – Режим доступу: https://www.ijcrt.org/papers/IJCRT2512181.pdf (accessed: 14.01.2026).
17. Best Python Frameworks for Data Science, AI, & Web Development – Anaconda – Режим доступу: https://www.anaconda.com/topics/python-frameworks

(дата звернення: 14.01.2026).

18. Evolution of Web Development Frameworks - IJRASET – Режим доступу: https://www.ijraset.com/research-paper/evolution-of-web-development-frameworks (дата звернення: 14.01.2026).

19. Python Framework Guide 2026: Best Frameworks for Every Use Case – Softjourn – Режим доступу: https://softjourn.com/insights/best-python-frameworks (дата звернення: 14.01.2026).

20. Python Frameworks for Microservices Development - Nevina Infotech – Режим доступу: https://www.nevinainfotech.com/blog/python-frameworks-for-microservices-development (дата звернення: 14.01.2026).

21. Research on the Effectiveness of Using the Python Language for Creating Cyber Security and Information Protection Applications - R Discovery – Режим доступу: https://discovery.researcher.life/article/research-on-the-effectiveness-of-using-the-python-language-for-creating-cyber-security-and-information-protection-applications/4f084616c3f33c279e285380d6b2f860 (дата звернення: 14.01.2026).

22. WSGI vs ASGI: The Crucial Decision Shaping Your Web App's Future in 2025 – Режим доступу: https://dev.to/leapcell/wsgi-vs-asgi-the-crucial-decision-shaping-your-web-apps-future-in-2025-3pcd (дата звернення: 14.01.2026).

23. How Do Python Framework APIs Evolve? An Exploratory Study – Режим доступу: https://sqlab-sustech.github.io/files/paper/SANER2020-preprint.pdf (дата звернення: 14.01.2026).

24. Benchmarking the performance of Python web frameworks – ResearchGate – Режим доступу: https://www.researchgate.net/publication/396039112_Benchmarking_the_performance_of_Python_web_frameworks (дата звернення: 14.01.2026).

25. Comprehensive Research on Python, Django, and the Power of Web Development Libraries – ijarsct – Режим доступу: https://www.ijarsct.co.in/Paper26357.pdf (дата звернення: 14.01.2026).

26. Django vs Flask: The Best Python Web Framework in 2025? – Cloudways – Режим доступу: https://www.cloudways.com/blog/django-or-flask/ (дата звернення: 14.01.2026).

27. Тези доповідей VII Міжнародної науково-практичної конференції «Інформаційні технології в освіті, науці і техніці» (ІТОНТ-2024), (Черкаси, 23-24 травня 2024 р.) [Електронний ресурс]. Черкаси : ЧДТУ, 2024. 349 с. – Режим доступу: https://itest.chdtu.edu.ua/Conference-Proceedings-ITEST-2024_25_06.pdf (дата звернення: 14.01.2026).

28. Application of Modern Python-type Information Systems in the Modeling of Ukrainian Agricultural Enterprises by Clusterization Method – ResearchGate – Режим доступу: https://www.researchgate.net/publication/396391870_Application_of_Modern_Python-type_Information_Systems_in_the_Modeling_of_Ukrainian_Agricultural_Enterprises_by_Clusterization_Method (дата звернення: 14.01.2026).

29. A Python Framework for Crop Yield Estimation Using Sentinel-2 Satellite Data – MDPI – Режим доступу: https://www.mdpi.com/2673-4834/6/1/15 (дата звернення: 14.01.2026).

30. Part 2 - A quick history of python web programming – mleue – Режим доступу: https://mleue.com/posts/quick-history-of-python-web-programming/ (дата звернення: 14.01.2026).

31. A Short History of Python Web Frameworks – Режим доступу: https://www.python-summit.ch/recordings/sps23_nafiul_islam_a_short_history_of_python_web_frameworks/sps23_nafiul_islam_a_short_history_of_python_web_frameworks.pdf (дата звернення: 14.01.2026).

32. WebFrameworks - Python Wiki – Режим доступу: https://wiki.python.org/moin/WebFrameworks (дата звернення: 14.01.2026).

33. Best Python Web Frameworks for Data Scientists: A Comprehensive Overview – Режим доступу: https://www.dasca.org/world-of-data-science/article/best-python-web-frameworks-for-data-scientists-a-comprehensive-overview (дата звернення: 14.01.2026).

34. Django vs. Flask: Which is Better for Web Developers? - Noble Desktop – Режим доступу: https://www.nobledesktop.com/blog/django-vs-flask (дата звернення: 14.01.2026).

35. FastAPI in 2025: Why 38% of Python Developers Are Switching | byteiota – Режим доступу: https://byteiota.com/fastapi-in-2025-why-38-of-python-developers-are-switching/ (дата звернення: 14.01.2026).

36. Python Application Servers in 2025: From WSGI to Modern ASGI Solutions – DeployHQ – Режим доступу: https://www.deployhq.com/blog/python-application-servers-in-2025-from-wsgi-to-modern-asgi-solutions (дата звернення: 14.01.2026).

37. Ultimate guide to FastAPI library in Python – Deepnote – Режим доступу: https://deepnote.com/blog/ultimate-guide-to-fastapi-library-in-python (дата звернення: 14.01.2026).

38. Python in the Backend in 2025: Leveraging Asyncio and FastAPI for High-Performance Systems - Nucamp Bootcamp – Режим доступу: https://www.nucamp.co/blog/coding-bootcamp-backend-with-python-2025-python-in-the-backend-in-2025-leveraging-asyncio-and-fastapi-for-highperformance-systems (дата звернення: 14.01.2026).

39. FastAPI for Microservices: High-Performance Python API Design Patterns - Talent500 – Режим доступу: https://talent500.com/blog/fastapi-microservices-python-api-design-patterns-2025/ (дата звернення: 14.01.2026).

40. Ultimate guide to Python frameworks for building scalable microservices – Peerbits – Режим доступу: https://www.peerbits.com/blog/guide-to-python-frameworks-for-scalable-microservices.html (дата звернення: 14.01.2026).

41. Мікросервіси на Python: як створити ефективні сучасні додатки - EPAM Ukraine – Режим доступу: https://careers.epam.ua/blog/microservices-in-python-how-to-build-efficient-modern-apps (дата звернення: 14.01.2026).

42. Розробка мікросервісів на Python: найкращі практики – FoxmindEd – Режим доступу: https://foxminded.ua/rozrobka-mikroservisiv-na-python/ (дата звернення: 14.01.2026).

43. How Ukrainian web development companies are leading in technology trends? – Режим доступу: https://webbookstudio.com/articles/how-ukrainian-web-development-companies-are-leading-in-technology-trengs/ (дата звернення: 14.01.2026).

44. Python for Artificial Intelligence and Machine Learning – IJTSRD – Режим доступу: https://www.ijtsrd.com/papers/ijtsrd79847.pdf (дата звернення: 14.01.2026).

45. Leveraging Python in AI and Machine Learning: A Survey of Techniques and Educational Approaches in Software Engineering – ResearchGate – Режим

доступу: https://www.researchgate.net/publication/388852036_Leveraging_Python_in_AI_and_Machine_Learning_A_Survey_of_Techniques_and_Educational_Approaches_in_Software_Engineering (дата звернення: 14.01.2026).

46. Python for AI Development: Frameworks, Tools & Tips | The AI Journal – Режим доступу: https://aijourn.com/python-for-ai-development-frameworks-tools-tips/ (дата звернення: 14.01.2026).

47. Сучасні цифрові технології та інноваційні методики навчання: досвід, тенденції, перспективи. Матеріали IX Міжнародної науково-практичної інтернет-конференції (м. Тернопіль, 28 квітня, 2022), 234 с – Режим доступу: https://lib.iitta.gov.ua/id/eprint/730407/1/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA%20%D1%82%D0%B5%D0%B7_28%20%D0%BA%D0%B2%D1%96%D1%82%D0%BD%D1%8F%202022%20%D1%80..pdf (дата звернення: 14.01.2026).

48. Zhao Caifeng Method and multimodal framework for enhanced melanoma metastasis diagnosis. Vinnytsia. Vinnytsia National Technical University – 2025 – Режим доступу: https://ida.vntu.edu.ua/wp-content/uploads/2025/06/Dissertation_Zhao_Caifeng-1.pdf (дата звернення: 14.01.2026).

49. Evolution and Future Trends in Web Development: A Comprehensive Review – Режим доступу: https://www.researchgate.net/publication/389902743_Evolution_and_Future_Trends_in_Web_Development_A_Comprehensive_Review (дата звернення: 14.01.2026).

50. Comparative analysis of HTTP request handling in different Python frameworks – Режим доступу: https://www.researchgate.net/publication/397954720_Comparative_analysis_of_HTTP_request_handling_in_different_Python_frameworks дата звернення: 14.01.2026).

References

1. Django (web framework) [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/Django_(web_framework) (accessed Dec. 25, 2025).

2. Flask (web framework) [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/Flask_(web_framework) (Dec. 25, 2025).

3. FastAPI [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/FastAPI (accessed Dec. 25, 2025).

4. TensorFlow [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/TensorFlow (accessed Dec. 25, 2025).

5. PyTorch [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/PyTorch (accessed Dec. 25, 2025).

6. Selenium – History of Selenium [Online] // selenium.dev. – Available at: https://www.selenium.dev/about/history/ (accessed Dec. 25, 2025).

7. Robot Framework – The History of Robot Framework [Online] // blog.efficode.com. – Available at: http://blog.efficode.com/the-history-of-robot-framework-3077efc16419 (Dec. 25, 2025).

8. Red Hat – How Ansible works [Online] // redhat.com. – Available at: https://www.redhat.com/en/ansible-collaborative/how-ansible-works (Dec. 25, 2025).

9. Ansible (software) [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/Ansible_(software) (accessed Dec. 25, 2025).

10. Framework Training – What the 8th Python Developer Survey means for

you! [Online] (Sep. 24, 2025) – frameworktraining.co.uk. – Available at: https://www.frameworktraining.co.uk/news-insights/what-eighth-python-developer-survey-means (Dec. 25, 2025).

11. Integrate.io – Python ETL Framework Usage Trends — 15 Statistics Shaping Data Pipeline Development in 2026 [Online] (Sep. 26, 2025) – Available at: https://www.integrate.io/blog/python-etl-framework-usage-trends/ (Dec. 25, 2025).

12. Bravo C. – Gripped by Python: 5 reasons why Python is popular among cybersecurity professionals [Online] (Apr. 25, 2024) – WeLiveSecurity (ESET). Available at: https://www.welivesecurity.com/en/secure-coding/python-5-reasons-popular-cybersecurity-professionals/ (Dec. 25, 2025).

13. Python Software Foundation – tkinter — Python interface to Tcl/Tk [Online] (Python 3.14.2 Documentation). – Available at: https://docs.python.org/3/library/tkinter.html (Dec. 25, 2025).

14. Celery (software) [Online] // Wikipedia. – Available at: https://en.wikipedia.org/wiki/Celery_(software) (accessed Dec. 25, 2025).

15. Fabric – Welcome to Fabric! [Online] // fabfile.org. – Available at: https://www.fabfile.org/ (Dec. 25, 2025).

16. Python As The Backbone Of Modern And Emerging Technologies - IJCRT.org – Режим доступу: https://www.ijcrt.org/papers/IJCRT2512181.pdf (accessed: 14.01.2026).

17. Best Python Frameworks for Data Science, AI, & Web Development – Anaconda – Available at: https://www.anaconda.com/topics/python-frameworks (accessed: 14.01.2026).

18. Evolution of Web Development Frameworks - IJRASET – Available at: https://www.ijraset.com/research-paper/evolution-of-web-development-frameworks (accessed: 14.01.2026).

19. Python Framework Guide 2026: Best Frameworks for Every Use Case – Softjourn – Available at: https://softjourn.com/insights/best-python-frameworks (accessed: 14.01.2026).

20. Python Frameworks for Microservices Development - Nevina Infotech – Available at: https://www.nevinainfotech.com/blog/python-frameworks-for-microservices-development (accessed: 14.01.2026).

21. Research on the Effectiveness of Using the Python Language for Creating Cyber Security and Information Protection Applications - R Discovery – Available at: https://discovery.researcher.life/article/research-on-the-effectiveness-of-using-the-python-language-for-creating-cyber-security-and-information-protection-applications/4f084616c3f33c279e285380d6b2f860 (accessed: 14.01.2026).

22. WSGI vs ASGI: The Crucial Decision Shaping Your Web App's Future in 2025 – Available at: https://dev.to/leapcell/wsgi-vs-asgi-the-crucial-decision-shaping-your-web-apps-future-in-2025-3pcd (accessed: 14.01.2026).

23. How Do Python Framework APIs Evolve? An Exploratory Study – Available at: https://sqlab-sustech.github.io/files/paper/SANER2020-preprint.pdf (accessed: 14.01.2026).

24. Benchmarking the performance of Python web frameworks – ResearchGate – Available at: https://www.researchgate.net/publication/396039112_Benchmarking_the_performance_of_Python_web_frameworks (accessed: 14.01.2026).

25. Comprehensive Research on Python, Django, and the Power of Web Development Libraries – ijarsct – Available at:

https://www.ijarsct.co.in/Paper26357.pdf (accessed: 14.01.2026).

26. Django vs Flask: The Best Python Web Framework in 2025? – Cloudways – Available at: https://www.cloudways.com/blog/django-or-flask/ (accessed: 14.01.2026).

27. Tezy dopovidei VII Mizhnarodnoi naukovo-praktychnoi konferentsii «Informatsiini tekhnolohii v osviti, nautsi i tekhnitsi» (ITONT-2024), (Cherkasy, 23-24 travnia 2024 r.) [Elektronnyi resurs]. Cherkasy : ChDTU, 2024. 349 c. – Available at: https://itest.chdtu.edu.ua/Conference-Proceedings-ITEST-2024_25_06.pdf (accessed: 14.01.2026).

28. Application of Modern Python-type Information Systems in the Modeling of Ukrainian Agricultural Enterprises by Clusterization Method – ResearchGate – Available at: https://www.researchgate.net/publication/396391870_Application_of_Modern_Python-type_Information_Systems_in_the_Modeling_of_Ukrainian_Agricultural_Enterprises_by_Clusterization_Method (accessed: 14.01.2026).

29. A Python Framework for Crop Yield Estimation Using Sentinel-2 Satellite Data – MDPI – Available at: https://www.mdpi.com/2673-4834/6/1/15 (accessed: 14.01.2026).

30. Part 2 - A quick history of python web programming – mleue – Available at: https://mleue.com/posts/quick-history-of-python-web-programming/ (accessed: 14.01.2026).

31. A Short History of Python Web Frameworks – Available at: https://www.python-summit.ch/recordings/sps23_nafiul_islam_a_short_history_of_python_web_frameworks/sps23_nafiul_islam_a_short_history_of_python_web_frameworks.pdf (accessed: 14.01.2026).

32. WebFrameworks - Python Wiki – Available at: https://wiki.python.org/moin/WebFrameworks (accessed: 14.01.2026).

33. Best Python Web Frameworks for Data Scientists: A Comprehensive Overview – Available at: https://www.dasca.org/world-of-data-science/article/best-python-web-frameworks-for-data-scientists-a-comprehensive-overview (accessed: 14.01.2026).

34. Django vs. Flask: Which is Better for Web Developers? - Noble Desktop – Available at: https://www.nobledesktop.com/blog/django-vs-flask (accessed: 14.01.2026).

35. FastAPI in 2025: Why 38% of Python Developers Are Switching | byteiota – Available at: https://byteiota.com/fastapi-in-2025-why-38-of-python-developers-are-switching/ (accessed: 14.01.2026).

36. Python Application Servers in 2025: From WSGI to Modern ASGI Solutions – DeployHQ – Available at: https://www.deployhq.com/blog/python-application-servers-in-2025-from-wsgi-to-modern-asgi-solutions (accessed: 14.01.2026).

37. Ultimate guide to FastAPI library in Python – Deepnote – Available at: https://deepnote.com/blog/ultimate-guide-to-fastapi-library-in-python (accessed: 14.01.2026).

38. Python in the Backend in 2025: Leveraging Asyncio and FastAPI for High-Performance Systems - Nucamp Bootcamp – Available at: https://www.nucamp.co/blog/coding-bootcamp-backend-with-python-2025-python-in-the-backend-in-2025-leveraging-asyncio-and-fastapi-for-highperformance-systems

(accessed: 14.01.2026).

39. FastAPI for Microservices: High-Performance Python API Design Patterns - Talent500 – Available at: https://talent500.com/blog/fastapi-microservices-python-api-design-patterns-2025/ (accessed: 14.01.2026).

40. Ultimate guide to Python frameworks for building scalable microservices – Peerbits – Available at: https://www.peerbits.com/blog/guide-to-python-frameworks-for-scalable-microservices.html (accessed: 14.01.2026).

41. Mikroservisy na Python: yak stvoryty efektyvni suchasni dodatky - EPAM Ukraine – Available at: https://careers.epam.ua/blog/microservices-in-python-how-to-build-efficient-modern-apps (accessed: 14.01.2026).

42. Rozrobka mikroservisiv na Python: naikrashchi praktyky – FoxmindEd – Available at: https://foxminded.ua/rozrobka-mikroservisiv-na-python/ (accessed: 14.01.2026).

43. How Ukrainian web development companies are leading in technology trends? – Available at: https://webbookstudio.com/articles/how-ukrainian-web-development-companies-are-leading-in-technology-trengs/ (accessed: 14.01.2026).

44. Python for Artificial Intelligence and Machine Learning – IJTSRD – Available at: https://www.ijtsrd.com/papers/ijtsrd79847.pdf (accessed: 14.01.2026).

45. Leveraging Python in AI and Machine Learning: A Survey of Techniques and Educational Approaches in Software Engineering – ResearchGate – Available at: https://www.researchgate.net/publication/388852036_Leveraging_Python_in_AI_and_Machine_Learning_A_Survey_of_Techniques_and_Educational_Approaches_in_Software_Engineering (accessed: 14.01.2026).

46. Python for AI Development: Frameworks, Tools & Tips | The AI Journal – Available at: https://aijourn.com/python-for-ai-development-frameworks-tools-tips/ (accessed: 14.01.2026).

47. Suchasni tsyfrovi tekhnolohii ta innovatsiini metodyky navchannia: dosvid, tendentsii, perspektyvy. Materialy IX Mizhnarodnoi naukovo-praktychnoi internet-konferentsii (m. Ternopil, 28 kvitnia, 2022), 234 c – Available at: https://lib.iitta.gov.ua/id/eprint/730407/1/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA%20%D1%82%D0%B5%D0%B7_28%20%D0%BA%D0%B2%D1%96%D1%82%D0%BD%D1%8F%202022%20%D1%80..pdf (accessed: 14.01.2026).

48. Zhao Caifeng Method and multimodal framework for enhanced melanoma metastasis diagnosis. Vinnytsia. Vinnytsia National Technical University – 2025 – Available at: https://ida.vntu.edu.ua/wp-content/uploads/2025/06/Dissertation_Zhao_Caifeng-1.pdf (accessed: 14.01.2026).

49. Evolution and Future Trends in Web Development: A Comprehensive Review – Available at: https://www.researchgate.net/publication/389902743_Evolution_and_Future_Trends_in_Web_Development_A_Comprehensive_Review (accessed: 14.01.2026).

50. Comparative analysis of HTTP request handling in different Python frameworks – Available at: https://www.researchgate.net/publication/397954720_Comparative_analysis_of_HTTP_request_handling_in_different_Python_frameworks (accessed: 14.01.2026).

Надійшла до редакції 15.01.2026, розглянута на редколегії 00.00.2026

 ISSN 2071-1077(print)
ISSN 2663-2411(online

Research on the Evolution of Python Frameworks and Their Impact on the Development of Information Technology

This article analyzes the development of the Python framework ecosystem and evaluates its impact on various fields of information technology. The study explores the transition from early CGI protocols to modern asynchronous architectures based on ASGI, which ensure high performance in high-load systems. It identifies patterns regarding the influence of architectural decisions (monolithic and microservices) on the stability, scalability, and security of information systems. The focus is placed on categories of web frameworks (Django, Flask, FastAPI, etc.), machine learning and deep learning frameworks (TensorFlow, PyTorch), test automation tools (Selenium, Robot Framework), DevOps tools (Ansible, Fabric), and toolsets for API development and graphical user interfaces.

The article details the key evolutionary stages of these frameworks, their application areas (software development, data science, cloud computing, automation, cybersecurity, system integration), and examples of their impact on the IT industry. According to Python community surveys for 2024–2025, the leading web frameworks are FastAPI (38%), Django (35%), and Flask (34%), while the top ML libraries are scikit-learn (68%), PyTorch (66%), and TensorFlow (49%). For instance, Django (created in 2003–2005) was designed for the rapid creation of complex, database-driven websites, whereas Flask (first released in 2010) became an immensely popular microframework used by companies like Pinterest and LinkedIn. TensorFlow (2015) and PyTorch (2016) facilitated the rapid expansion of artificial intelligence tools. In the field of automation, Selenium (since 2004) and Robot Framework (since 2005) became de facto standards for web and system interface testing, while DevOps tools such as Ansible (2012) and Fabric significantly simplified system configuration and software deployment. Furthermore, Python frameworks for scientific computing (Pandas, NumPy, SciPy) and ML tasks provided the core instruments for data science. Overall, the study examines current trends and practical examples of Python framework applications in IT, demonstrating their importance in accelerating development and implementing innovations.

Key words: Python, frameworks, API evolution, WSGI, ASGI, microservices architecture, Django, Flask, TensorFlow, PyTorch, Selenium, Robot Framework, Ansible, Fabric, software development, data science, cloud computing, automation, cybersecurity.

Відомості про автора:

Столяренко Тетяна Леонідівна – кандидат педагогічних наук, доцент кафедри Економічної кібернетики і системного аналізу Харківського національного економічного університету ім. Семена Кузнеця. Електронна пошта: tatsto1978@gmail.com, телефон 0975218476, ORCID: 0000-0002-7202-316X.

About the author:

Stoliarenko Tetyana – Candidate of Pedagogical Sciences, Associate Professor of the Department of Economic Cybernetics and System Analysis Simon Kuznets Kharkiv National University of Economics Email: tatsto1978@gmail.com, Phone: 0975218476, ORCID: 0000-0002-7202-316X