УДК 004.8
Тарас М. - ст. гр. СН-43
Тернопільський національний технічний університет імені Івана Пулюя

СУЧАСНА ВЕБРОЗРОБКА З ВИКОРИСТАННЯМ NEXT.JS, REACT ТА DJANGO REST

Науковий керівник: к.т.н, доцент Дмитроца Л.П.

Taras M.
Ternopil Ivan Puluj National Technical University

MODERN WEB DEVELOPMENT USING NEXT.JS, REACT AND DJANGO REST

Supervisor: PhD, Associate Professor Dmytrotsa L.P.

Ключові слова: технології, веброзробка, фреймворк
Keywords: technologies, web-development, framework

У світі цифрових технологій вебзастосунки стали не просто засобом комунікації чи презентації інформації – вони перетворилися на фундамент бізнесу, освіти, медицини та навіть особистої взаємодії. Зі зростанням попиту на швидкі, зручні та масштабовані рішення постає питання щодо вибору інструментів для створення сучасного вебсайту чи сервісу?

Розробники мають у своєму розпорядженні десятки фреймворків, бібліотек і платформ. Однак серед цього розмаїття окремо вирізняються три популярні технології: React, Next.js та Django REST Framework. Усі вони зарекомендували себе як потужні інструменти, але кожна має свою роль, підхід і сферу найкращого застосування.

Порівняння цих технологій ґрунтується не лише на технічних характеристиках, а й на практичних аспектах застосування в реальних проєктах. Зокрема, береться до уваги основні аспекти роботи, їх переваги та обмеження, а також взаємодія між собою для створення швидких, масштабованих та ефективних рішень.

React – це JavaScript-бібліотека, створена компанією Meta у 2013 році для побудови користувацьких інтерфейсів. Із моменту свого запуску React швидко набув популярності завдяки своїй простоті, гнучкості та активній спільноті [1]. У сучасній веброзробці він став майже стандартом для створення динамічних фронтенд-рішень.

Головна ідея React – компонентність. Усі частини інтерфейсу – кнопки, форми, списки – реалізуються як незалежні, багаторазово використовувані компоненти. Це значно спрощує розробку, масштабування та підтримку великих застосунків. Наприклад, якщо в застосунку є форма реєстрації, її можна створити як окремий компонент і надалі використовувати в кількох місцях без дублювання коду.

Однією з ключових інновацій React стала концепція віртуального DOM – внутрішнього уявлення сторінки, яке дозволяє мінімізувати кількість змін у реальному DOM браузера. Це покращує продуктивність застосунку, особливо на мобільних пристроях або при повільному інтернет-з’єднанні.

React орієнтований на побудову SPA – односторінкових застосунків, які працюють без повного перезавантаження сторінки. Користувач взаємодіє з додатком швидко, немовби перебуває в мобільному застосунку, тоді як нові дані підвантажуються динамічно у фоновому режимі.

Next.js – це фреймворк на базі React, створений компанією Vercel, який значно розширює можливості розробника у створенні сучасних вебзастосунків. Якщо React – це конструктор інтерфейсів, то Next.js – це повноцінний інструмент для розробки повнофункціональних застосунків, який охоплює як фронтенд, так і частину серверної логіки [2].

Однією з головних переваг Next.js є підтримка різних типів рендерингу [3]:

- SSR (Server-Side Rendering) – сторінки генеруються на сервері при кожному запиті, що покращує SEO та перше завантаження.
- SSG (Static Site Generation) – сторінки попередньо рендеряться під час збірки, що дає неймовірну швидкість на фронтенді.
- ISR (Incremental Static Regeneration) — гібридна модель, яка дозволяє оновлювати вже згенеровані сторінки частково, без повного перезбирання сайту.

Завдяки цим можливостям Next.js особливо добре підходить для сайтів з публічним контентом, блогів, маркетплейсів та платформ, де важливі як продуктивність, так і пошукова оптимізація.

На відміну від React, де роутинг налаштовується вручну, Next.js використовує автоматичну маршрутизацію, що базується на структурі файлів у папці pages. Це спрощує логіку навігації й економить час при розробці.

Next.js дозволяє створювати серверні API-ендпоінти прямо в межах проєкту, без необхідності окремого бекенду. Це зручно для невеликих сервісів, адмін-панелей або прототипів. Однак для масштабних рішень все ж доцільно використовувати повноцінний бекенд (наприклад, Django REST).

Next.js фактично є містком між фронтендом і сервером. Він дозволяє створювати гібридні застосунки, де деякі частини генеруються на сервері, інші – в браузері, а логіка бекенду може частково жити всередині самого проєкту. Але що робити, коли потрібен потужний, незалежний бекенд, який обробляє великі обсяги даних, має розвинену авторизацію, взаємодіє з базою даних і надає API? Тут у гру вступає Django REST Framework.

Django REST Framework (DRF) – це розширення до одного з найпопулярніших Python-фреймворків Django, яке дозволяє швидко й ефективно створювати RESTful API [4]. Якщо React і Next.js відповідають за інтерфейс та логіку на стороні користувача, то Django REST забезпечує серверну частину застосунку – взаємодію з базами даних, авторизацію, бізнес-логіку та надання даних у зручному форматі (наприклад, JSON).

Одна з головних переваг DRF – це використання зрілої, перевіреної часом архітектури Django. У ньому чітко визначено моделі, маршрутизацію, контролери (view) і шаблони, що дозволяє швидко стартувати з проєктом та дотримуватись зрозумілої структури.

DRF доповнює Django спеціалізованими класами для створення API: Serializer, APIView, ViewSet, Router та іншими. Це дозволяє гнучко керувати даними, перетворюючи об'єкти моделей у формат, придатний для фронтенду (наприклад, React/Next.js).

Django REST Framework має вбудовану підтримку:

- Аутентифікації через токени, сесії, OAuth2.
- Права доступу до ресурсів (permissions).
- Обмеження запитів (rate limiting).
- Документації API через Swagger або ReDoc.

Це робить його придатним для розробки великомасштабних та безпечних проєктів, які обробляють тисячі запитів щодня – від інтернет-магазинів до освітніх платформ і медичних систем.

Відмінності та переваги кожної з розглянутих технологій подані в таблиці 1.

Таблиця 1 – Порівняння ключових характеристик React, Next.js та Django REST

Критерій	**React**	**Next.js**	**Django REST Framework**
Тип технології	JavaScript-бібліотека	React-фреймворк (full-stack frontend)	Python-фреймворк для REST API
Роль у проєкті	Побудова інтерфейсу (UI)	Побудова повного фронтенду з SSR/SSG	Створення серверного API
Мова програмування	JavaScript / TypeScript	JavaScript / TypeScript	Python
Рендеринг	Клієнтський (CSR)	Клієнтський + серверний (CSR/SSR/SSG/ISR)	Немає інтерфейсу, лише API
SEO-оптимізація	Слабка	Висока завдяки SSR/SSG	Не застосовується
Маршрутизація (роутинг)	Вручну через React Router	Автоматично за структурою файлів	Через URLConf (ручна конфігурація)
Підтримка бекенду	Ні (потребує стороннього API)	Частково (через API-роути)	Так (повноцінний бекенд)
Підключення до БД	Ні	Частково (через ORM або API)	Так (ORM Django, SQL/NoSQL)
Простота старту	Висока	Середня	Середня/висока (для Python-розробників)
Крива навчання	Полога	Помірна	Помірна до крутої
Готовність до масштабування	Потребує налаштувань	Висока (оптимізації "з коробки")	Висока (чітка архітектура та безпека)
Ідеальний випадок застосування	SPA, UI-бібліотеки	SEO-дружні сайти, повні фронтенд-рішення	Побудова надійного API для мобільних та вебзастосунків

Кожна з технологій виконує свою роль у сучасному вебстеку:

- React – ідеальний для створення інтерфейсів і швидкої розробки компонентів.
- Next.js – потужний фреймворк для створення повноцінних фронтенд-застосунків із серверним рендерингом.
- Django REST – чудовий вибір для бекенду з акцентом на безпеку, структуру і масштабованість.

У сучасній веброзробці ефективність часто визначається не лише якістю окремих інструментів, а й тим, наскільки вдало вони поєднуються між собою. Саме тому комбінація React та Next.js на фронтенді та Django REST Framework на бекенді стала популярною архітектурною моделлю для створення продуктивних, масштабованих та зручних у підтримці вебзастосунків.

Одна з головних переваг такої зв’язки – можливість розділити застосунок на два незалежні шари:

- Фронтенд (React та Next.js) – відповідає за інтерфейс, взаємодію з користувачем, маршрутизацію, відображення даних.

- Бекенд (Django REST) – забезпечує логіку бізнес-процесів, обробку запитів, збереження даних, автентифікацію.

Це дозволяє різним командам працювати паралельно, використовувати переваги кожної технології, а також масштабувати або оновлювати окремі частини системи незалежно.

Обмін даними між фронтендом і бекендом відбувається через REST API, побудований за допомогою Django REST Framework. Фронтенд надсилає запити до серверу (наприклад, отримати список курсів або надіслати замовлення), а бекенд відповідає даними у форматі JSON.

Цей підхід має кілька ключових переваг:

- Універсальність – API може використовуватись не лише вебклієнтом, а й мобільними застосунками.
- Безпека – чіткий контроль доступу та автентифікації.
- Гнучкість – легко змінювати інтерфейс без змін у логіці сервера.

Приклад типового стеку:

- Frontend: Next.js + React, з можливим використанням Tailwind CSS або Material UI для стилізації.
- Backend: Django + Django REST Framework + PostgreSQL або MySQL.
- Інтеграція: Axios або Fetch API для HTTP-запитів; JWT або сесії для автентифікації; Stripe або PayPal для платіжних систем (при потребі).

Таке поєднання технологій дозволяє розробляти інтерактивні, швидкі, масштабовані вебзастосунки, де фронтенд забезпечує зручний користувацький досвід, а бекенд гарантує надійну обробку даних. Успішна інтеграція React/Next.js з Django REST – це вже не просто тренд, а ефективна архітектурна практика, яка відповідає вимогам сучасного ринку.

Висновок. Аналіз сучасних вебтехнологій React, Next.js та Django REST Framework показав, що їх поєднання створює потужний технологічний стек для розробки масштабованих, продуктивних та зручних у використанні вебзастосунків. React забезпечує гнучкість інтерфейсів, Next.js додає можливості серверного рендерингу та оптимізації, а Django REST виконує роль надійного бекенду з API-доступом до даних. Така інтеграція дозволяє ефективно реалізовувати як прості, так і комплексні проєкти, відповідаючи вимогам сучасного цифрового середовища.

Література

1. 9 Reasons Why React is Still Popular in 2024. KOMODO. URL: https://www.komododigital.co.uk/insights/9-reasons-why-react-is-still-popular-in-2021/ (дата звернення: 04.04.2025).
2. Bhati A. Next JS vs React: Which Framework Should You Choose For Web Development?. TRooInbound. 10.06.2024. URL: https://www.trooinbound.com/blog/next-js-vs-react/ (дата звернення: 08.04.2025).
3. Aditya Kumar Tiwari. Understanding CSR, SSR, SSG, and ISR: A Next.js Perspective. Medium. 10.07.2023. URL: https://medium.com/design-bootcamp/understanding-csr-ssr-ssg-and-isr-a-next-js-perspective-fcaf36686de6 (дата звернення: 10.04.2025).
4. Junaidul Islam. The Perfect Pair: Why Next.js and Django REST Framework are Ideal for Full-Stack Web Development. DEV. 04.09.2024. URL: https://dev.to/junaaid96/the-perfect-pair-why-nextjs-and-django-rest-framework-are-ideal-for-full-stack-web-development-4a02 (дата звернення: 10.04.2025).