УДК 004.43:004.451.86
DOI https://doi.org/10.32782/tnv-tech.2025.2.10

СУЧАСНІ ФРЕЙМВОРКИ JAVASCRIPT І МАЙБУТНЄ JAVASCRIPT ЯК ПОВНОЦІННОЇ МОВИ ПРОГРАМУВАННЯ

Козак Є. Б. *– магістр у галузі комп'ютерних наук, розробник програмного забезпечення, інженер-програміст*
GAN Inc., Лондон, Велика Британія
ORCID ID: 0000-0002-8342-2609

У цьому дослідженні детально описано сучасні фреймворки JavaScript, JavaScript розглядається як повноцінна мова програмування. Структуровано генезис JavaScript та відокремлено перспективи на майбутнє. Здійснено порівняння з відомими мовами програмування сьогодення та описано переваги JavaScript перед іншими мовами. Наголошується, що JavaScript став однією з найпопулярніших мов програмування завдяки своїй можливості виконувати складні задачі на стороні клієнта та сервера. JavaScript розглядається, як динамічна, об'єктно-орієнтована прототипна мова програмування, яка часто використовується для створення інтерактивних веб-застосунків і сайтів. Фреймворки JavaScript є потужними інструментами, що перетворюють веб-сторінки на інтерактивні та привабливі. До загальних переваг сучасних JavaScript-фреймворків варто віднесено: прискорення розробки, структурованість коду, компонентний підхід, масштабованість, кросплатформеність, спільнота та ресурси, тестування, підтримка асинхронності. Фреймворки спрощують реалізацію асинхронних запитів, що є критично важливим для сучасних веб-додатків, які використовують API і працюють з даними в режимі реального часу.

Підкреслено, що JavaScript володіє всіма ознаками сучасної мови програмування: це об'єктно-орієнтоване програмування – підтримує об'єкти та класи; функціональне програмування – підтримує функції такі як перші класи та λ-вираження; асинхронність – можливість роботи з асинхронними операціями за допомогою колбеків та async / await; статична типізація. Зазначається, що хоча JavaScript динамічно типізований, завдяки TypeScript та іншим інструментам можлива статична типізація.

JavaScript невпинно еволюціонує і знаходить застосування в різних сферах, від веб-розробки до серверної логіки, що підтверджує його статус повноцінної мови програмування.

Ключові слова: *мова програмування, JavaScript, фреймворк, переваги, генезис, інструмент, розробка, перспективи.*

Cozac E. B. The latest JavaScript frameworks and JavaScript as the main programming language of the future

This study describes in detail modern JavaScript frameworks, JavaScript is considered as a full-fledged programming language. The genesis of JavaScript is structured and future prospects are identified. A comparison is made with well-known programming languages of today and the advantages of JavaScript over other languages are described. It is emphasized that JavaScript has become one of the most popular programming languages due to its ability to perform complex tasks on the client and server sides. JavaScript is considered as a dynamic, object-oriented prototyping programming language, which is often used to create interactive web applications and sites. JavaScript frameworks are powerful tools that transform web pages into interactive and attractive ones. The general advantages of modern JavaScript frameworks include: development acceleration, code structure, component approach, scalability, cross-platform, community and resources, testing, asynchrony support. Frameworks simplify the implementation of asynchronous requests, which is critical for modern web applications that use APIs and work with data in real time.

It is emphasized that JavaScript has all the features of a modern programming language: it is object-oriented programming – supports objects and classes; functional programming – supports functions such as first classes and λ-expressions; asynchrony – the ability to work with asynchronous operations using callbacks and async / await; static typing. It is noted that although JavaScript is dynamically typed, static typing is possible thanks to TypeScript and other tools.

JavaScript is constantly evolving and finds application in various areas, from web development to server-side logic, which confirms its status as a full-fledged programming language.

Key words: *programming language, JavaScript, framework, advantages, genesis, tool, development, prospects.*

Вступ. На початку свого існування JavaScript, як окреме впровадження призначалося задля розширення інтерактивності веб-сторінок, однак з часом, JavaScript вийшла на новий рівень. Цьому, у першу чергу, посприяла розробка сучасних фреймворків JavaScript. Саме вони сприяли перетворенню JavaScript на комплексну мову програмування, яка займає лідерство за обсягом коду та є стандартом для серйозних проектів. Головними причинами актуальності JavaScript на протязі багатьох років є те, що JavaScript є основною мовою для веб-розробки, і більшість сайтів використовують її для інтерактивності та динамічності. З розвитком бібліотек і фреймворків, таких як React, Angular і Vue. js, JavaScript став основним інструментом для розробки сучасних веб-додатків. Завдяки Node. js, JavaScript може використовуватися не лише на клієнтській стороні, але й для серверної розробки, що забезпечує універсальність. Досить активна спільнота JavaScript розробників, велика кількість ресурсів, туторіалів, бібліотек, що дозволяє новим розробникам швидко навчатися та отримувати підтримку. Тенденції на кшталт Web Assembly, Progressive Web Apps, та використання JavaScript у мобільній розробці (наприклад, через React Native) свідчать про його максимальну актуальність в умовах сьогодення.

Аналіз останніх досліджень і публікацій. У науково-дослідницькому просторі останніх років спостерігається зростання уваги до проблематики розвитку JavaScript як повноцінної мови програмування та до аналізу ефективності фреймворків, що на її основі використовуються для створення сучасних веб-застосунків. У низці досліджень акцентовано увагу на визначенні популярних JavaScript-фреймворків та порівнянні їх функціональних можливостей і продуктивності. Зокрема, у роботі В. Башова, В. Стаценка та Д. Стаценка [1] окреслено перелік найбільш поширених фреймворків для побудови web-інтерфейсів, серед яких виокремлено React, Angular та Vue. Дослідниками здійснено емпіричний аналіз швидкодії цих рішень під час виконання операцій зі списком інтерактивних елементів, що дозволило встановити перевагу Angular у контексті часу повного формування сторінки порівняно з React та Vue.

У рамках іншого дослідження М. Горським, М. Огірком, І. Солтисом та співавторами [2] проведено систематизацію сучасних фреймворків, що застосовуються у веб-розробці, із поділом їх на дві основні категорії – CSS-фреймворки та JavaScript-фреймворки. Науковці підкреслюють, що при виборі JavaScript-фреймворків доцільно враховувати такі аспекти, як продуктивність і швидкодія, реактивність, розмір бібліотек, регулярність оновлень і підтримку спільноти, а також легкість вивчення та застосування. У дослідженні детально проаналізовано фреймворки React, Angular, Vue. js, Svelte, а також CSS-фреймворки Material-UI, Ant Design, Foundation та Bootstrap, з акцентом на їхню роль у прискоренні процесу розробки, забезпеченні якості та консистентності інтерфейсів.

Окрему увагу приділено питанням підвищення продуктивності JavaScript-додатків шляхом інтеграції з іншими мовами програмування та технологіями. Так, у дослідженні В. Козуба [3] представлено результати експериментального впровадження мови Rust у середовище JavaScript за допомогою механізмів WebAssembly та власних прив'язувань. Автором аргументовано показано, що застосування Rust сприяє суттєвому зниженню обчислювальних витрат, особливо в сценаріях паралельної обробки даних, а також дозволяє ефективно подолати архітектурні обмеження JavaScript.

Вагомим внеском у дослідження поточного стану та перспектив JavaScript-фреймворків є аналітична праця Е. Малохвія, В. Бугая, Г. Молчанова та О. Черних [4]. Автори здійснили порівняння сучасних рішень у сфері веб-розробки, спираючись на статистичні дані з таких джерел, як Google Trends, Stack Overflow Developer Survey, GitHub Stars, The State of JavaScript та популярні платформи пошуку роботи. Особливу увагу приділено аналізу популярних рішень React і Vue, а також визначенню їхніх переваг та недоліків.

Ефективність використання бібліотеки React детально досліджено у роботі О. Безверхого та О. Куценка [5]. Дослідники підкреслюють, що React є відкритим та масштабованим інструментом для побудови динамічних користувацьких інтерфейсів, що дозволяє створювати великі веб-застосунки зі змінними даними без необхідності перезавантаження сторінки. Водночас наголошується на широкому використанні й інших фреймворків і бібліотек, таких як Vue. js, Angular, Svelte, Jquery, Meteor та Backbone, кожна з яких має свої особливості й сфери застосування.

Аналіз наукових джерел засвідчує, що питанням розробки та використання JavaScript-фреймворків присвячено низку праць зарубіжних і вітчизняних науковців, серед яких Г. Галімуллін, Е. Гузуєва [6], Ф. Феррейра, Х. Боржес, М. Валенте [7], А. Шукла [8], Т. Амблер, Н. Клауд [9], А. Пано, Д. Граціотін, П. Абрагамссон [10], Д. Мокогінта, Д. Путрі, Ф. Ваттімена [11], К. Анді, Д. Рамірес, Г. Чанго [12], А. Карич, Н. Дурмич [13], Ї. Зоу, Ж. Чжай, Ч. Фан, Дж. Лю, Т. Чжен, Ж. Чен [14] та інші.

Водночас слід зазначити, що, незважаючи на наявність значної кількості праць, які торкаються окремих аспектів функціонування та використання JavaScript-фреймворків, комплексне дослідження перспектив розвитку JavaScript як універсальної мови програмування та системний аналіз ефективності фреймворків у контексті створення масштабованих і продуктивних веб-застосунків залишаються недостатньо опрацьованими. Більшість наявних досліджень фрагментарно висвітлюють окремі технічні характеристики або питання популярності інструментів, що ускладнює формування цілісного уявлення про еволюцію JavaScript-екосистеми. Отже, актуальним є проведення подальших теоретичних і прикладних досліджень, спрямованих на виявлення факторів, які визначають ефективність застосування JavaScript-фреймворків у різних галузях програмної інженерії, а також на прогнозування тенденцій розвитку цієї мови програмування в умовах зростання вимог до якості, продуктивності та безпеки веб-застосунків.

Метою роботи є дослідження сучасних фреймворків JavaScript і майбутнього JavaScript як повноцінної мови програмування.

Виклад основного матеріалу. Розроблений у 1995 році Брендоном Айком, як проста мова сценаріїв для вдосконалення веб-сторінок, JavaScript зазнав надзвичайного розвитку. Спочатку він вважався клієнтською мовою для додавання інтерактивності веб-сайтам. Проте стандартизація ECMAScript у 1997 році під керівництвом ECMA International зіграла ключову роль у формуванні його розвитку та стандартизації його функцій. У наступні роки JavaScript зазнав значного розвитку. Вийшли нові версії ECMAScript, зокрема ES3 (1999) та ES5 (2009), які додали багато нових можливостей і покращили мову. У 2015 році вийшов ECMAScript 2015 (ES6), який став значним оновленням, що внесло ряд великих удосконалень, таких як стрілочні функції, класи, модулі та нові методи для масивів, що змінило спосіб розробки на JavaScript. Специфікація ES6 значно покращила мову програмування, включаючи функції зі стрілками, шаблонні літерали та оголошення let і const. Ці інновації виявилися дуже корисними для підвищення читабельності

коду та зручності обслуговування. Крім того, запровадження модулів ES6 створило стандартизований підхід до організації коду JavaScript, сприяння модульності та полегшення повторного використання коду.

Значення стандартизації ES6 і ECMAScript для еволюційної траєкторії JavaScript неможливо підкреслити. Вищевказані досягнення дозволили досягти кросбраузерної сумісності та надали розробникам уніфікований набір функцій і можливостей. Таким чином, модулі ES6 покращують організацію коду та сприяють його підтримці. Впровадження цих стандартизацій зіграло значну роль у розвитку JavaScript, що призвело до його перетворення на більш надійну та зручну мову для розробників.

Еволюцію JavaScript можна простежити через його різні версії, від ECMAScript 1 до останньої версії ECMAScript 2021, кожна з яких містить нові функції та вдосконалення. Ці зміни поступово перетворили JavaScript на більш надійну та універсальну мову, здатну вирішувати складні завдання програмування. Сьогодні JavaScript є основною мовою для веб-розробки, з величезною екосистемою бібліотек та фреймворків (таких як React, Angular, та Vue. js), що робить його надзвичайно популярним серед розробників.

Сучасні фреймворки JavaScript зіграли центральну роль в еволюції мови. React, розроблений і підтримуваний Facebook, набув широкої популярності завдяки своїй компонентній архітектурі та віртуальному DOM, що значно покращує розробку інтерактивного інтерфейсу користувача. Компонентний дизайн React і віртуальний DOM значно покращили взаємодію з користувачем. Це можна пояснити хорошим представленням компонентів інтерфейсу користувача. Зручність React і підтримка спільноти розробників сприяють його широкому застосуванню. Angular, що підтримується Google, пропонує комплексну структуру, яка спрощує розробку інтерфейсу, надаючи структуру для створення динамічних веб-додатків. Фреймворк Angular, розроблений Google, демонструє виняткову майстерність у створенні динамічних веб-додатків. Впровадження залежностей і двостороннє зв'язування даних підвищують продуктивність розробників і покращують організацію коду. Vue. js відомий завдяки своїм прогресивним атрибутам, легко зрозумілій траєкторії навчання та універсальному розгортанню. Можливість поступового включення Vue. js дозволяє розробникам поступово інтегрувати його функції у свої проекти. Цей атрибут робить Vue. js привабливим варіантом для людей з різним ступенем кваліфікації в розробці програмного забезпечення.

Ці фреймворки демократизували веб-розробку, надавши розробникам інструменти, які сприяють повторному використанню коду, зручності обслуговування та масштабованості. Вони також зробили внесок у активну екосистему бібліотек з відкритим кодом і ресурсів, керованих спільнотою, які продовжують просувати JavaScript вперед.

Значним досягненням стало просування JavaScript у напрямку комплексного розвитку стеків. Поява Node. js, яка була запущена в 2009 році, призвела до помітної революції в реалізації серверного JavaScript. Впровадження Node. js полегшило виконання JavaScript на сервері, встановивши цілісну парадигму розробки, яка використовує єдину мову програмування для операцій на стороні клієнта та на стороні сервера.

Переваги цієї комбінації значні, використання цієї методології дає змогу обмінюватися кодом між зовнішніми та внутрішніми елементами, що призводить до підвищення ефективності процесів розробки та зменшення частоти мовних переходів, необхідних розробникам.

Node.js пропонує середовище виконання, яке дає розробникам змогу використовувати JavaScript для бекенд-розробки, що полегшує розробку повного стекового рішення.

Процес уніфікації здійснив оптимізацію процесів розробки, спрощення спільного використання коду між інтерфейсом і сервером, а також підвищив ефективність розробки.

Крім того, слід зазначити, що Node Package Manager (npm) став значним репозиторієм для пакетів, бібліотек і модулів JavaScript. Зазначена платформа відіграє значну роль у підвищенні можливості повторного використання коду та розширенні адаптивності JavaScript як основної мови програмування. Наявність диверсифікованої та розгалуженої екосистеми npm значно пришвидшила цикли розробки та мінімізувала потребу в тиражуванні вже існуючих рішень.

Сучасні фреймворки JavaScript суттєво змінили ландшафт повноцінної розробки, надаючи можливість розробникам створювати масштабовані та надійні програми як на стороні клієнта, так і на стороні сервера. Проте є деякі ключові нововведення та функції, які сприяють ролі JavaScript.

Так, наприклад серверний Javascript Node. js, дозволяє розробникам використовувати JavaScript на сервері. Він також дає змогу розробляти повний стек за допомогою однієї мови програмування. Така уніфікація також спрощує робочі процеси розробки та сприяє повторному використанню коду між сервером і клієнтом.

Асинхронне програмування у JavaScript корисно при розробці з повним стеком, коли деякі асинхронні завдання обробляють декілька клієнтів, операції з базою даних і запити є звичайним.

JavaScript вважається єдиною мовою, яку можна використовувати для повноцінної розробки, а її широке впровадження та стабільне вдосконалення позиціонують її як найкращу мову програмування. У порівнянні з іншими мовами повного стеку, JavaScript має переваги у послідовності, бо дозволяє розробникам використовувати ту саму мову як на стороні сервера, так і на стороні клієнта. Це зведе до мінімуму перемикання контексту та підвищує зручність обслуговування коду. Спільнота цієї мови є великою та сприяє максимізації ресурсів, підтримці, керованій спільнотою, та документації, яка може сприяти спільному середовищу для розробників. Ще одним важливим фактором виступає універсальність. JavaScript можна використовувати для різноманітних додатків від невеликих проектів до додатків корпоративного рівня. Завдяки своїй універсальності та адаптованості це сприяє її популярності в різноманітних сценаріях розвитку.

Оскільки Node. js забезпечує продуктивне та масштабоване середовище виконання для серверної сторони, яке забезпечує ефективну обробку одночасних запитів і підтримує розробку високопродуктивних програм, JavaScript має перевагу у середовищі виконання.

Аналізуючи сучасні фреймворки JavaScript варто наголосити, що сильною стороною є стислий синтаксис для написання функцій, що покращує читабельність коду; літерали шаблону спрощують інтерполяцію рядків.

Асинхронне програмування спрощує обробку всіх видів асинхронних операцій і робить код читабельним і придатним для обслуговування порівняно з підходами на основі зворотного виклику.

Модулі (ES6 і CommonJS) формують представлення модульного коду, це пов'язано із запровадженням вбудованої підтримки модулів, застосованої в ES6, а формат Common JS полегшує генерацію модульного коду, що підвищує зручність обслуговування та багаторазове використання.

Код API Promise надає пропозицію щодо забезпечення чистого способу обробки асинхронних операцій шляхом мінімізації зворотного виклику та покращення обробки помилок.

WebAssembly (WASM) – це платформа веб-розробки з відкритим вихідним кодом, що дозволяє запускати код, скомпільований з різних мов програмування, у веб-браузерах. Це не мова програмування, а формат байт-коду. Wasm складається з трьох основних програм: «операційної системи», редактора та однієї або декількох оболонок. Ця функція дозволяє запускати код зі швидкістю, близькою до рідної, і виконувати завдання, що потребують інтенсивної продуктивності, у браузері та на сервері.

Використання функції Biglnt вводить підтримку цілих чисел довільної точності, і вони використовуються для таких сценаріїв, які вимагають великих цілих значень.

JavaScript має можливість працювати на стороні клієнта та серверів. Таким чином, це забезпечує уніфікований мовний стек і сприяє повторному використанню коду, а також спрощує співпрацю між інтерфейсними та внутрішніми розробниками.

З появою WebAssembly у JavaScript його можливості для критично важливих завдань розширені та пов'язані з перевищенням продуктивності інших мов у таких сценаріях.

JavaScript пережив значну метаморфозу, еволюціонувавши від клієнтської мови сценаріїв до потужної повноцінної мови програмування.

Еволюція розвитку JavaScript вказує на значний вплив стандартизації ECMAScript і подальшого запуску ES6. Ці досягнення не тільки розкривають стандартизацію мови, але й пропонують інші характеристики та вдосконалення, які значно підвищують читабельність і зручність обслуговування коду. Використання модулів ES6 полегшило структурування коду, призвело до підвищення модульності та полегшення повторного використання коду. Впровадження цих стандартизацій мало вирішальне значення для перетворення JavaScript на більш потужну та зручнішу мову програмування.

Неможливо переоцінити вплив сучасних фреймворків JavaScript, таких як React, Angular і Vue. js, на веб-розробку. Ці фреймворки суттєво вплинули на сферу розробки інтерфейсу, надаючи програмістам надійні інструменти для проектування. Поліпшення взаємодії з користувачем можна пояснити використанням React компонентної архітектури та віртуального DOM. Гнучка архітектура Angular і прогресивний характер Vue. js призначені для задоволення різноманітних вимог розробників у різних доменах.

Включення JavaScript у розробку з повним стеком, сприяло появі Node. js, що призвело до покращення робочих процесів розробки та зменшення складності в управлінні декількома мовами програмування для розробок інтерфейсу та бек-енду. Керований подіями та неблокуючий механізм введення-виведення, який використовується Node. js, відіграв важливу роль у розробці серверних програм, які демонструють виняткові характеристики продуктивності. Можливості JavaScript як повноцінної мови було посилено завдяки включенню широкого спектру пакетів і бібліотек через Node Package Manager.

До загальних переваг сучасних JavaScript-фреймворків варто віднести:

1. Прискорення розробки. Фреймворки забезпечують готові рішення для поширених задач, що дозволяє розробникам зекономити час на реалізацію базових функцій.

2. Структурованість коду. Фреймворки пропонують чітку архітектуру і шаблони проектування (MVC, MVVM), що допомагає структурувати код і робить його більш зрозумілим та підтримуваним.

3. Компонентний підхід. Багато фреймворків, як-от React та Vue. js, використовують компонентний підхід, що дозволяє розробникам створювати повторно використовувані компоненти, спрощуючи розробку та тестування.

4. Масштабованість. Середовище фреймворків дозволяє легко розширювати застосунки, додаючи нові функції без значних змін базового коду.

5. Кросплатформеність. Багато сучасних фреймворків (наприклад, React Native) дозволяють розробляти кросплатформенні рішення, що працюють на різних пристроях і платформах, включаючи мобільні телефони та настільні комп'ютери.

6. Спільнота та ресурси. Популярні фреймворки мають великі спільноти, що забезпечує підтримку, документацію, плагіни та бібліотеки для швидкого вирішення різних завдань.

7. Тестування. Багато фреймворків надають вбудовані можливості для тестування, що полегшує перевірку коду на наявність помилок і покращує якість готового продукту.

8. Підтримка асинхронності. Фреймворки спрощують реалізацію асинхронних запитів, що є критично важливим для сучасних веб-додатків, які використовують API і працюють з даними в режимі реального часу.

Таким чином, використання JavaScript-фреймворків дозволяє оптимізувати процес розробки, знижує ймовірність помилок і покращує обслуговування кодової бази.

Висновки. Загалом, дослідження сучасних фреймворків JavaScript та його майбутнього показало, як JavaScript розвинувся від мови сценаріїв для операцій на стороні клієнта до повноцінної мови програмування, здатної керувати всіма частинами розробки програмного забезпечення за допомогою постійних творчих інновацій. JavaScript є життєво важливим у веб-розробці через стандартизовані мовні протоколи, нові фреймворки та можливості повноцінної розробки. Завдяки своїй гнучкості, адаптивності та використанню мова має великий потенціал.

Компанії шукають шляхи підвищення адаптивності та сумісності JavaScript у відповідь на зростаючу потребу в ефективних підходах до розробки веб-додатків. Очікується, що і можливості, і варіанти використання JavaScript значно збільшаться. Існування активної спільноти розробників і безперервне вдосконалення фреймворків і інструментів гарантують незмінну важливість JavaScript у веб-розробці. Послідовне зростання JavaScript у динамічній сфері веб-розробки є яскравим прикладом потужності інновацій та адаптивності.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. Башовий В., Стаценко В., Стаценко Д. Визначення швидкості роботи сучасних фреймворків для створення web-інтерфейсів. *Technologies and Engineering*. 2023. P. 9-16. DOI: 10.30857/2786-5371.2022.4.1

2. Горський М., Огірко М., Солтис І., Дуболазов О., Ушенко О., В. Морфлюк-Щур, Слоцька Л. Сучасні фреймворки для створення користувацького інтерфейсу в технологіях електронних видань. *Технологія і техніка друкарства*. 2024. P. 93-100. DOI: 10.20535/2077-7264.3(85).2024.293209

3. Kozub V. Code optimization opportunities in the JavaScript ecosystem with Rust. *LatIA*. 2024. № 2. P. 68. DOI: 10.62486/latia202468

4. Малохвій Є. В., Бугай В. В., Молчанов Г. В., Черних О. В. Аналітичний огляд та порівняння сучасних JavaScript рішень для розробки веб-додатків. *Сис-*

теми управління, навігації та зв'язку. Збірник наукових праць. 2021. № 4. P. 55–58. DOI: 10.26906/SUNZ.2021.4.055

5. Безверхий О., Куценко О. Ефективність застосування бібліотеки react. *INFORMATION TECHNOLOGY AND SOCIETY*. 2022. P. 13-19. DOI: 10.32689/maup.it.2022.2.2

6. Galimullin N., Guzueva E. Developing interactive web applications using modern javascript frameworks. *Ekonomika i upravlenie: problemy, resheniya*. 2024. № 11/7. P. 167–178. DOI: 10.36871/ek.up.p.r.2024.11.07.017

7. Ferreira F., Borges H., Valente M. On the (un-)adoption of JavaScript front-end frameworks. *Software: Practice and Experience*. 2021. P. 52. DOI: 10.1002/spe.3044

8. Shukla A. An Innovative Solution to Manage Projects using Different Javascript Frameworks. *Journal of Artificial Intelligence & Cloud Computing*. 2023. P. 1–3. DOI: 10.47363/JAICC/2023(2)220

9. Ambler T., Cloud N. JavaScript frameworks for modern web dev. Berkeley, CA: Apress, 2015. 485 p. DOI: https://doi.org/10.1007/978-1-4842-0662-1

10. Pano A., Graziotin D., Abrahamsson P. Factors and actors leading to the adoption of a JavaScript framework. *Empirical Software Engineering*. 2018. № 23. DOI: 10.1007/s10664-018-9613-x

11. Mokoginta D., Putri D., Wattimena F. Developing Modern JavaScript Frameworks for Building Interactive Single-Page Applications. *International Journal Software Engineering and Computer Science (IJSECS)*. 2024. № 4. P. 484-496. DOI: 10.35870/ijsecs.v4i2.2831

12. Andi K., Ramirez D., Chango G. Comparativa de frameworks más usados de JavaScript mediante la creación de una aplicación web. Mikarimin. *Revista Científica Multidisciplinaria*. 2024. № 10. P. 81–100. DOI: 10.61154/mrcm.v10i3.3245

13. Karic A., Durmic N. Comparison of JavaScript Frontend Frameworks – Angular, React, and Vue. *International Journal of Innovative Science and Research Technology (IJISRT)*. 2024. P. 1383-1390. DOI: 10.38124/ijisrt/IJISRT24JUN600

14. Zou Y., Zhai J., Fang C., Liu J., Zheng T., Chen Z. Mutation-Based Deep Learning Framework Testing Method in JavaScript Environment. *ArXiv* : website. 2024. DOI: 10.48550/arXiv.2409.14968

REFERENCES:

1. Bashovyi, V., Statsenko, V., & Statsenko, D. (2023). Vyznachennia shvydkosti roboty suchasnykh freimvorkiv dlia stvorennia web-interfeisiv [Determination of the performance speed of modern frameworks for creating web interfaces]. *Technologies and Engineering – Technologies and Engineering*, 9–16. https://doi.org/10.30857/2786-5371.2022.4.1 [In Ukrainian].

2. Bezzverkhyi, O., & Kutsenko, O. (2022). Efektyvnist zastosuvannia biblioteky React [Efficiency of using the React library]. *INFORMATION TECHNOLOGY AND SOCIETY – Information Technology and Society*, 13–19. https://doi.org/10.32689/maup.it.2022.2.2 [In Ukrainian].

3. Ferreira, F., Borges, H., & Valente, M. (2021). On the (un-)adoption of JavaScript front-end frameworks. *Software: Practice and Experience*, 52. https://doi.org/10.1002/spe.3044 [In English].

4. Galimullin, N., & Guzueva, E. (2024). Developing interactive web applications using modern JavaScript frameworks. *Ekonomika i upravlenie: problemy, resheniya – Economics and Management: Problems, Solutions*, 11(7), 167–178. https://doi.org/10.36871/ek.up.p.r.2024.11.07.017 [In English].

5. Gorskyi, M., Ohirko, M., Soltys, I., Dubolazov, O., Ushchenko, O., Morfliuk-Shchur, V., & Slotska, L. (2024). Suchasni freimvorky dlia stvorennia korystvuatskoho interfeisu v tekhnolohiiakh elektronnykh vydan [Modern frameworks for creating user interfaces in electronic publishing technologies]. *Tekhnolohiia i tekhnika*

drukarstva–TechnologyandTechniqueofPrinting,93–100.https://doi.org/10.20535/2077-7264.3(85).2024.293209 [In Ukrainian].

6. Karic, A., & Durmic, N. (2024). Comparison of JavaScript Frontend Frameworks – Angular, React, and Vue. *International Journal of Innovative Science and Research Technology (IJISRT)*, 1383–1390. https://doi.org/10.38124/ijisrt/IJISRT24JUN600 [In English].

7. Kozub, V. (2024). Code optimization opportunities in the JavaScript ecosystem with Rust. *LatIA*, 2, 68. https://doi.org/10.62486/latia202468 [In English].

8. Malokhviĭ, Ye. V., Buhaĭ, V. V., Molchanov, H. V., & Chernykh, O. V. (2021). Analitychnyi ohliad ta porivniannia suchasnykh JavaScript rishen dlia rozrobky veb-dodatkiv [Analytical review and comparison of modern JavaScript solutions for web application development]. *Systemy upravlinnia, navihatsii ta zviazku. Zbirnyk naukovykh prats – Control, Navigation and Communication Systems. Collection of Scientific Works*, 4, 55–58. https://doi.org/10.26906/SUNZ.2021.4.055 [In Ukrainian].

9. Mokoginta, D., Putri, D., & Wattimena, F. (2024). Developing modern JavaScript frameworks for building interactive single-page applications. *International Journal Software Engineering and Computer Science (IJSECS)*, 4, 484–496. https://doi.org/10.35870/ijsecs.v4i2.2831 [In English].

10. Pano, A., Graziotin, D., & Abrahamsson, P. (2018). Factors and actors leading to the adoption of a JavaScript framework. *Empirical Software Engineering*, 23. https://doi.org/10.1007/s10664-018-9613-x [In English].

11. Shukla, A. (2023). An innovative solution to manage projects using different JavaScript frameworks. *Journal of Artificial Intelligence & Cloud Computing*, 1–3. https://doi.org/10.47363/JAICC/2023(2)220 [In English].

12. Ambler, T., & Cloud, N. (2015). *JavaScript frameworks for modern web dev*. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-0662-1 [In English].

13. Andi, K., Ramirez, D., & Chango, G. (2024). Comparativa de frameworks más usados de JavaScript mediante la creación de una aplicación web. *Mikarimin. Revista Científica Multidisciplinaria*, 10, 81–100. https://doi.org/10.61154/mrcm.v10i3.3245 [In Spanish].

14. Zou, Y., Zhai, J., Fang, C., Liu, J., Zheng, T., & Chen, Z. (2024). Mutation-based deep learning framework testing method in JavaScript environment. *ArXiv*. https://doi.org/10.48550/arXiv.2409.14968 [In English].