УДК 004.896
DOI https://doi.org/10.32689/maup.it.2022.2.2

Олександр БЕЗВЕРХИЙ
доктор фізико-математичних наук, професор, професор кафедри інформаційних систем і технологій, Національний транспортний університет, вул. М. Омеляновича-Павленка, 1, Київ, індекс 01010 (o_bezver@ukr.net)

ORCID: 0000-0002-0834-6335
Scopus Author ID: 6603638908, 14067133500

Олександр КУЦЕНКО
аспірант кафедри інформаційних систем і технолоій, Національний транспортний університет, вул. М. Омеляновича-Павленка, 1, Київ, індекс 01010 (alexkutsenko95@gmail.com)

ORCID: 0000-0003-0047-4874

Oleksandr BEZVERKY
Doctor of Physical and Mathematical Sciences, Professor, Professor at the Department of Information Systems and Technologies, National Transport University, 1 M. Omelyanovicha-Pavlenka str., Kyiv, postal code 01010 (o_bezver@ukr.net)

Oleksandr KUTSENKO
Postgraduate Student at the Department of Information Systems and Technologies, National Transport University, 1 M. Omelyanovicha-Pavlenka str., Kyiv, postal code 01010 (alexkutsenko95@gmail.com)

Бібліографічний опис статті: Безверхий, О., Куценко, О. (2022). Ефективність застосування бібліотеки React. *Інформаційні технології та суспільство*, 2 (4), 13–19. DOI: https://doi.org/10.32689/maup.it.2022.2.2

Bibliographic description of the article: Bezverkhyi, O., Kutsenko, O. (2022). Efektyvnist zastosuvannia biblioteky React. [The efficiency of using the React library]. *Informatsiini tekhnolohii ta suspilstvo – Information technology and society*, 2(4), 13–19. DOI: https://doi.org/10.32689/maup.it.2022.2.2

ЕФЕКТИВНІСТЬ ЗАСТОСУВАННЯ БІБЛІОТЕКИ REACT

Для відображення додатків існують ряд бібліотек та фреймворків. Частіше застосовують React, Vue js, Angular, Svelte, Jquery, Meteor, Backbone. React JS – це бібліотека JavaScript з відкритим кодом, яка використовується спеціально для побудови користувальницьких інтерфейсів. Основна перевага React JS полягає в тому, що він масштабований, простий та швидкий. React дозволяє розробникам створювати великі веб-застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. Як бібліотеку інтерфейсу користувача React найчастіше використовують разом з іншими бібліотеками, такими як Redux. Ефективність застосування бібліотеки React полягає в:

1. Односторонній передачі даних. Властивості передаються в рендерер компоненту, як властивості html тегу. Компонент не може напряму змінювати властивості, що йому передані, але може їх змінювати через callback функції. Такий механізм називають «властивості донизу, події нагору».

2. Віртуальному DOM. React підтримує віртуальний DOM, а не покладається виключно на DOM браузера. Це дозволяє бібліотеці визначити, які частини DOM змінилися, порівняно (diff) зі збереженою версією віртуального DOM, і таким чином визначити, як найефективніше оновити DOM браузера.

Ключові слова: *розробка веб-додатків, Реакт, діджиталізація, комп'ютерні науки.*

THE EFFICIENCY OF USING THE REACT LIBRARY

Contains a number of libraries and frameworks to display applications. More often Back React, Vuejs, Angular, Svelte, Jquery, Meteor, Backbone. React JS is an open source JavaScript library used specifically to build user interfaces. The main advantage of React JS is that it is large-scale, simple and fast. React allows retailers to create large web applications that use data that changes over time without reloading pages. Which React user interface library is used in conjunction with other libraries, such as Redux. The effectiveness of the React library is in:

1. One-way data transmission. Properties are passed to the component renderer as html tag properties. A component cannot directly change the properties passed to it, but can change them through callback functions. This mechanism is called "properties down, events up".

2. Virtual DOM. React supports virtual DOM and does not rely solely on browser DOM. This allows the library to determine which parts of the DOM have changed, compared (diff) with the saved version of the virtual DOM, and thus determine how best to update the browser DOM.

Key words: *Web application development, React, digitalization, computer science.*

Вступ. Актуальність статті полягає у дослідженні інноваційної бібліотеки React що широко застосовується для відображення додатків. Метою роботи є дослідження всіх аспектів та переваг данної бібліотеки над усіма іншими фреймворками, та її інтеграцію з різними фронт-енд технологіями.

На сьогодні існують бібліотеки та фреймворки для відображення додатків, серед них частіше застосовують React, Vue js, Angular, Svelte, Jquery, Meteor, Backbone. Вони мають у собі ряд нових підходів та технологій для ефективного створення, тестування та роботи додатків. Авто-тестування коду спрощує процес тестування та економить час.

Сучасний розвиток бібліотек розпочинається за появи Jquery. Вона вирішувала питання універсального коду, адже раніше в кожному браузері був свій JS, що працювала через маніпулювання з DOM. Проте у бібліотеки Jquery не було чіткої структури для написання повноцінного додатку та її код виглядав як спагетті.

З 2010 року розпочалося створення повноцінних фреймворків для написання клієнтського додатку. Першою бібліотекою був Backbone, а за ним Angular js від компанії Google. У цих бібліотеках з'явився клієнтський роутинг та відповідно можна було створювати односторінковий додаток (Single page application).

Ще одним предстанвиком того часу був Ember який використовував ще HTML шаблонізатор Handlebars.

З появи першого фреймворку, та в процесі ускладнення написання клієнтської частини додатку, з'являється необхідність у розширенні області веб розробки фахівцями з фронтенду.

Front-end розробник – це фахівець в області веб-розробки, в завдання якого входить проектування користувацьких інтерфейсів для сайтів або додатків, технічні рішення в області проектування веб-інтерфейсів, що забезпечують зручність користування веб-ресурсом. Спеціалісти даної області повинні досконало знати HTML, CSS, JavaScript, основи SEO, розуміти термін «кросбраузерність», «юзабіліті» та завжди застосовувати їх на практиці. Головне завдання Front-end розробника – Зробити веб-додаток так, щоб він швидко працював, був надійний та привабливий.

React JS – це бібліотека JavaScript з відкритим кодом, яка використовується спеціально для побудови користувальницьких інтерфейсів. Розробляється та підтримується компанією Facebook з 2013 року, також є моїм основним робочим інструментом. Зазвичай використовується для односторінкових програм. Він використовується для обробки всіх переглядів програми для будь-яких веб-або мобільних додатків. ReactJS також використовується для повторного використання компонентів інтерфейсу. React дозволяє розробникам створювати веб-програми, які можуть змінювати ваші дані, не завантажуючи вашу сторінку. Основна перевага React JS полягає в тому, що він масштабований, простий та швидкий. Це також відповідає виду в шаблоні MVC. Зазвичай він виступає комбінацією бібліотек або фреймворків JavaScript.

1. Аналіз основних конкурентів

В ході дослідження всіх популярних фрон-енд фрейморків було виділено наступні:

Angular (зазвичай так називають фреймворк Angular 13 або Angular 2+, тобто вищі версії) – написаний на TypeScript front-end фреймворк з відкритим кодом, який розробляється під керівництвом Angular Team у компанії Google, а також спільнотою приватних розробників та корпорацій. Angular – це AngularJS, який був переосмислений та перероблений тією ж командою розробників.

Всередині фреймворку реалізовано:

– Модульність.
– Анімації.
– Машрутизація.
– Робота з бекендом.
– Зберігання/обробка/відображення данних.

Згальний вигляд архітектури Ангуляру:

Перш за все, варто зазначити, що Angular застосунки пишуться на TypeScript, а не на чистому JavaScript. TypeScript – мова програмування, розробленна компанією Microsoft у 2012 році, як мова яка розширює можливості Javascript та надає статичної типізації, компіляції та інше.

Архітектура Angular складається з: Module, Component, Template, Service, Router, Pipe, Directives.

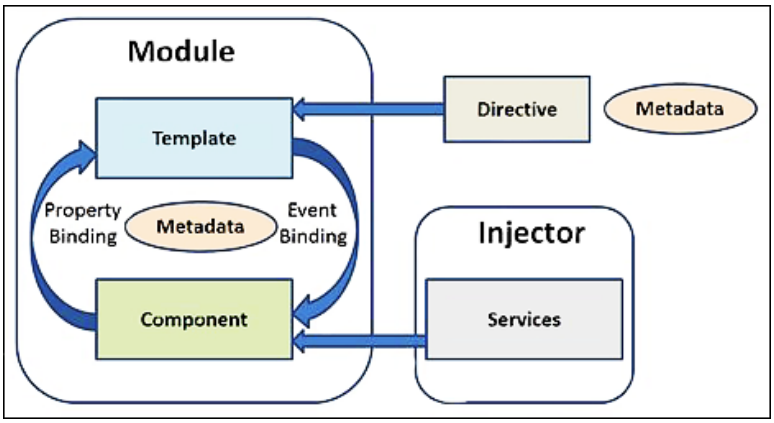

Рис. 1. Схема роботи Angular

Модулі (Module) – структурні одиниці застосунку, які інкапсулюють певну логіку. В Angular це структури, які зберігають певні компоненти, директиви та сервіси, об'єднані певною логікою. Прикладом може слугувати профіль користувача, модуль для написання листа, огляд списку листів тощо.

Компоненти (Component) – typescript клас, який зберігає дані та логіку відображення цих даних у шаблоні (представленні). Шаблон тісно пов'язаний з компонентом. Дані з компонента можна з легкістю відображати у шаблоні, використовуючи спеціальний синтаксис. Компонент також може «знімати» дані з шаблона та отримувати їх безпосередньо у скрипті.

Шаблон **(**Template**)** – фрагмент html-коду з додаванням спеціального синтаксису. Він дозволяє впроваджувати в шаблон дані з компонента без використання innerHTML та подібних методів. Шаблон прописується у компоненті та є частиною його конфігурації.

Сервіс **(**Service**)** в Angular являє собою typescript класи, які виконують задачі, пов'язані з отриманням, зберіганням та обробкою даних. Наприклад, логування, перетворення даних для подальшої передачі у компонент, звернення до backend та ін. На відміну від компонентів та директив сервіси не працюють з представленнями (шаблонами) напряму.

Задачі сервісів:

– надання даних застосунку. Сервіс сам може зберігати дані у пам'яті або, з метою отримання даних, звертатися до якогось джерела даних, наприклад, до сервера;

– сервіс може організувати канал взаємодії між окремими компонентами застосунку;

– сервіс може інкапсулювати бізнес-логіку, різноманітні обчислювальні задачі, задачі з логування, які краще виносити поза компоненти. Таким чином, код компонентів буде зосереджений, безпосередньо, на роботі з представленням. До того ж, можемо розв'язати проблему повторення коду, якщо нам знадобиться виконати одну й ту саму задачу у різних компонентах і класах.

Роутер (Router) – маршрутизатор, який призначений для переходу між екранами з метою відображення різного контенту. Іншими словами, коли в адресному рядку браузера у вас змінюється фрагмент URL, маршрутизатор відстежує ці зміни та завантажує ту або іншу частину застосунку.

Директиви та Пайпи – більш специфічні конструкції, які простіше продемонструвати у коді, ніж описати словами.

Vue.js

Це фреймворк, який знайшов баланс між обмеженнями та гнучкістю. Його ядро вирішує передусім задачі представлення даних, тому він легко інтегрується в наявні проекти поступово. Проте підходить для створення повноцінних SPA (Single-Page-Application) застосунків, адже має повний набір функціоналу.

Завдяки хорошій документації, низькому рівню входу та невеликому розміру досить активно завойовує серця розробників.

Він має CLI, маршрутизацію, як Angular, використовує Virtual DOM і має досить швидкий час розробки, як React.

Та все ж таки він має менш гнучкий компонентний підхід, ніж React, а керування життєвим циклом, що відбувається «під капотом» у великих проєктах, може стати проблемою через неочевидність і важкість відлагодження.

З цього можна зробити висновок, що Vue.js круто підходить для розробки більшості вебзастосунків, але якщо треба робити надто складне відображення, звернутися варто до React.

Svelte

Не можу не згадати ще одного цікавого кандидата у світі фронтенду – Svelte. Це принципово новий підхід до розробки фронтенду, адже коли типові фреймворки завантажуються вам у браузер і тоді починають свою роботу, то Svelte є компілятором, що переводить код, написаний з використанням власного синтаксису, в елегантний і оптимізований чистий JS-код. Без Virtual DOM, без абстракцій, лише чистий низькорівневий JS.

І здавалось би, в чому тут переваги, ми втрачаємо всі бонуси, які дають нам фреймворки…

Насправді велика перевага в тому, що нам не потрібно з собою в браузер користувача тягнути сам фреймворк, немає додаткових навантажень, для обрахування Virtual DOM, а також збірник сміття JS може більш ефективно підчищати пам'ять, яку використовує програма.

Варто зазначити, що для великих застосунків, де 25–50 чи навіть 170 КБ фреймворку це лише 1–2 % від усього розміру застосунку, а Virtual DOM дає більше користі, ніж використовує ресурсів, переваги Svelte є несуттєвими.

З усім тим він активно набирає популярність і може стати новим подихом у світі фронтенду. Але вакансій на цю мить дуже мало.

2. React та його переваги

React (**React.js**, **ReactJS**) – відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. Розробляється Facebook, Instagram і спільнотою індивідуальних розробників, на сьогодні актуальною версією є 18.

React дозволяє розробникам створювати великі веб-застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. Його мета полягає в тому, щоб бути швидким, простим, масштабованим. React обробляє тільки користувацький інтерфейс у застосунках. Це відповідає *видові* у шаблоні модель-вид-контролер (MVC), і може бути використане у поєднанні з іншими JavaScript бібліотеками або в великих фреймворках MVC, таких як AngularJS. Він також може бути використаний з React на основі надбудов, щоб піклуватися про частини без користувацького інтерфейсу побудови веб-застосунків. Як бібліотеку інтерфейсу користувача React найчастіше використовують разом з іншими бібліотеками, такими як Redux.

В даний час React використовують Khan Academy, Netflix, Yahoo, Airbnb, Sony, Atlassian та інші.

Бібліотеку створено Джорданом Волком (Jordan Walke), програмістом з Facebook. Автор працював над проектом під впливом XHP, фреймворку HTML для PHP. 2011-го року реліз з'явився у новинах Facebook, за рік – у соціальній мережі Instagram.Також фреймворк був представлений як проект з відкритим початковим кодом на конференції розробників JSConf US, що проходила у Сполучених Штатах у травні 2013 року. На конференції React.js Conf, влаштовану Фейсбуком у березні 2015-го, проект було представлено як відкрите програмне забезпечення.

Особливості:

Одностороння передача даних. Властивості передаються в рендерер компоненту, як властивості html тегу. Компонент не може напряму змінювати властивості, що йому передані, але може їх змінювати через callback функції. Такий механізм називають «властивості донизу, події нагору».

Віртуальний DOM. React підтримує віртуальний DOM, а не покладається виключно на DOM браузера. Це дозволяє бібліотеці визначити, які частини DOM змінилися, порівняно (diff) зі збереженою версією віртуального DOM, і таким чином визначити, як найефективніше оновити DOM браузера.Таким чином програміст працює зі сторінкою, вважаючи що вона оновлюється вся, але бібліотека самостійно вирішує які компоненти сторінки треба оновити.

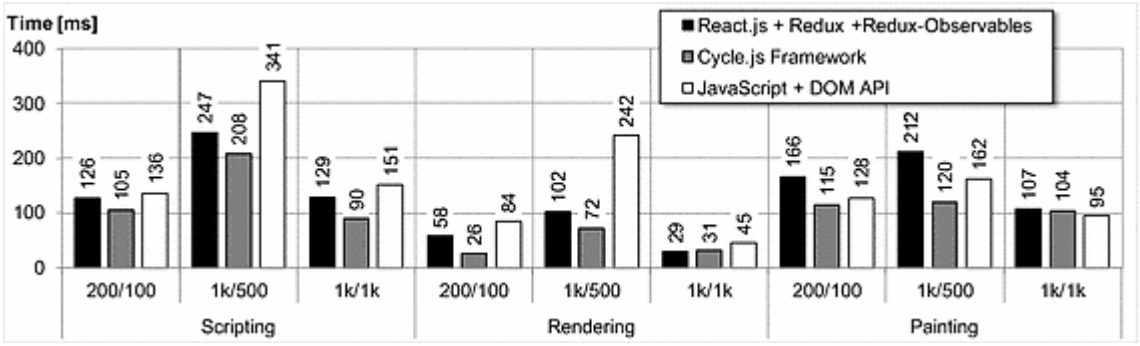

Рис. 2. Порівняння швидкості Virtual Dom з DOM

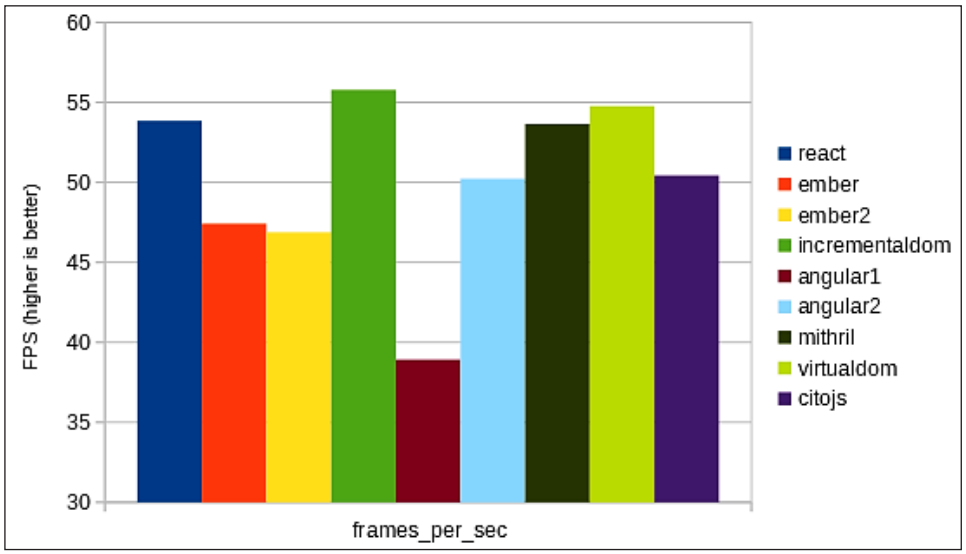

Рис. 3. Порівняння швидкості React з іншими бібліотеками та фреймворками

JSX – це розширення синтаксису JavaScript. Він подібний до мови шаблонів, але наділений всіма можливостями JavaScript. JSX компілюється у виклики React.createElement(), котрі повертають прості об'єкти JavaScript, що називаються «React-елементи».

React DOM використовує стиль camelCase для найменування властивостей замість звичайних імен HTML-атрибутів. Наприклад, tabindex в JSX перетворюється в tabIndex. Атрибут class записується як className, оскільки в JavaScript слово class є зарезервованим.

Не лише рендеринг HTML в браузері. React використовують не лише для рендерингу HTML в браузері. Наприклад, Facebook має динамічні графіки які рендеряться в теги <canvas>, Netflix та PayPal використовують ізоморфне завантаження для рендерингу ідентичного HTML на сервері та клієнті.

Методи життєвого циклу

Методи життєвого циклу – це визначена користувачем функціональність, що виконується протягом різних етапів життя компонента. Є методи, котрі доступні коли компонент створюється і вставляється в DOM (монтування), коли компонент оновлюється і коли компонент відмонтовується і видаляється з DOM. Наприклад:

– shouldComponentUpdate – це метод життєвого циклу, який каже Javascript оновити компонент, використовуючи логічні змінні;

– componentWillMount – це метод життєвого циклу, який каже Javascript налаштувати певні дані перед монтуванням компонентів (вставлення у віртуальний DOM);

– componentDidMount – це метод життєвого циклу, подібний до компонента WillMount, за винятком того, що він працює після методу render, і може використовуватися для додавання JSON-даних, а також для визначення властивостей та станів;

– render є найважливішим методом життєвого циклу, необхідним у будь-якому компоненті. Метод render – це те, що з'єднується з JSX і відображати власний JSX;

– починаючи з версії 16.8 можно використовувати Хуки для життєвого цикли.

Вкладені елементи

Кілька елементів на одному рівні повинні бути загорнутими в один елемент контейнера, наприклад елемент , або компонент <Fragment>, або повернутий як массив.

Атрибути

JSX надає ряд атрибутів елементів, призначених для відображення тих, що надаються у форматі HTML. Користувацькі атрибути також можуть бути передані компоненту. Всі атрибути будуть отримані компонентом як реквізит.

Props

Пропс – це вхідні дані React-компонента. Вони є даними, що передаються від батьківського компонента до дочірнього Запам'ятайте, що props призначені лише для читання. Не варто намагатися змінювати їх. Якщо вам потрібно змінити якесь значення у відповідь на ввід користувача чи відповідь сервера, використовуйте state (стан).

State

Компонент потребує state, коли якісь дані в ньому змінюються з часом. Наприклад, компоненту Checkbox може знадобитися isChecked у його стані, а компонент NewsFeed має відслідковувати fetchedPosts у своєму стані.

Найбільша різниця між state і props полягає в тому, що props передаються з батьківського компонента, а state керується самим компонентом. Компонент не може змінювати власні props, але може змінювати state. Кожна окрема частина змінних даних має бути під керуванням єдиного компоненту, що має її в своєму стані. Не намагайтесь синхронізувати стани між двома різними компонентами. Замість цього підійміть його до найближчого батьківського компонента і передайте його через пропси до кожного дочірнього компоненту.

Компоненти

React-компоненти – це маленькі, придатні для повторного використання частини коду, що повертають React-елемент для його відображення на сторінці. Найпростіший React-компонент – це звичайна функція JavaScript, що повертає React-елемент. Також, компоненти можуть бути классами ES6. Компоненти можуть бути контрольованими та неконтрольованими. У контрольованих компонентах е стан, а неконтрольовані просто відображають дані. Компоненти можна розбити на окремі частини залежно від їх функціональності і використовувати всередині інших компонентів. Компоненти можуть повертати інші компоненти, масиви, рядки і числа. Якщо якась частина вашого інтерфейсу використовується у кількох місцях (Button, Panel, Avatar) чи надто складна сама по собі, завжди є сенс винести її в незалежний компонент. Імена компонентів завжди мають починатися з великої літери (<Wrapper/>, а не <wrapper/>).

Узгодження

Коли пропси чи стан компонента змінюються, React порівнює тільки що повернутий і попередній відрендерений елемент та вирішує, чи потрібно оновлювати DOM. Якщо вони не рівні, то React здійснює оновлення DOM. Цей процес і називається «узгодження».

Fiber – це нова архітектура, що покладена в основу React 16, реліз якого у 2017 році. Велика частина коду була переписана з нуля. Основною метою було створення можливості для пріоритизації оновлень контенту. Також переписана система обробки помилок та усунуті деякі старі незручності, наприклад, необхідність обгортати декілька елементів в один кореневий елемент. Існуюче API, на щастя, майже не зачепили. Саме Fiber робить Реакт найкращим.

React не намагається надати повну «схему додатків». Він безпосередньо спрямований на побудову користувацьких інтерфейсів, і тому не включає в себе безліч інструментів, які деякі розробники вважають необхідними для створення програми. Це дозволяє вибрати будь-які бібліотеки, які розробник вважає за краще виконувати, щоб виконати певних завдань, таких як здійснення доступу до мережі або локальне зберігання даних.

Висновки. На початку становлення фронтенду була тільки одна бібліотека, але вона мала свої недоліки. Через це почали створюватися бібліотеки та фреймворки для написання повноцінних додатків. Найпершими булі Ангуляр та Backbone.

На сьогодні є багато бібліотек так фреймворків для створення додатків проте React розроблений компанією Facebook (Meta) є найкращим.

Він поєднує простоту, ефективність та легку інтеграцію з будь-якою бібліотекою.

Проаналізувавши бібліотеку Реакт можна зробити висновки що весь додаток складається з компонентів. Компоненти бувають класові та функціональні. В останніх версіях всі можливості класових компонентах доступні у функціональних через нові Хуки. З новою архітектурою Fiber з приоритизацією контенту Реакт стає однозначним фаворитом серед усіх бібліотек та фреймворків для створення додатків.

Проте його розробка триває і зовсім скоро він знову нас порадує своїми новими покращеннями.

Отже на сьогодні є багато засобів для створення додатків розроблені різними світовими компаніями, вони мають різну архітектуру та підходи, проте якщо ви хочете швидко та надійно створити додаток, то Реакт є найкращим для цього.

Список використаних джерел:

1. Офіційна документація React js. URL: https://uk.reactjs.org/
2. React 16: огляд нової архітекури Fiber. Евген Шеремет. URL: https://dou.ua/lenta/articles/react-fiber/
3. Огляд фреймворків JavaScript. Що, для чого і коли використовувати. URL: https://dou.ua/forums/topic/34739/
4. Angular (фреймворк). URL: https://ru.wikipedia.org/wiki/Angular_(фреймворк)
5. Які інтерфейсні фреймворки найкраще використовувати у 2022 році? Девід Карчевскі. URL: https://www.ideamotive.co/blog/best-frontend-frameworks

6. Відомі і ті, що ховаються в тіні: найпопулярніші фреймворки для фронтенд розробки 2020. URL: https://luxnet.io/uk/blog/popular-frameworks-for-front-end-development-2020
7. Фронтенд-2019: підсумки року. URL: https://senior.ua/articles/frontend2019-pdsumki-roku
8. React. URL:https://uk.wikipedia.org/wiki/React
9. Веб-застосунок. URL:https://uk.wikipedia.org/wiki/Веб-застосунок
10. Vue js. URL:https://ru.wikipedia.org/wiki/Vue.js

References:

1. Ofitsiina dokumentatsiia React js [Official documentation React js]. Retrived from: https://uk.reactjs.org/ [in Ukrainian]
2. React 16: ohliad novoi arkhitekury Fiber. Evhen Sheremet [React 16: overview new Fiber architecture. Evegen Sheremet]. Retrived from: https://dou.ua/lenta/articles/react-fiber/ [in Ukrainian]
3. Ohliad freimvorkiv JavaScript. Shcho, dlia choho i koly vykorystovuvaty [Overview JavaScript frameworks. What, for what use and when]. Retrived from: https://dou.ua/forums/topic/34739/ [in Ukrainian]
4. Angular (freimvork). Retrived from: https://ru.wikipedia.org/wiki/Angular_(фреймворк) [in Russian]
5. Yaki interfeisni freimvorky naikrashche vykorystovuvaty u 2022 rotsi? Devid Karchevski [Which interface frameworks are the best in 2022? David Karchewski]. Retrived from: https://www.ideamotive.co/blog/best-frontend-frameworks [in Ukrainian]
6. Vidomi i ti, shcho khovaiutsia v tini: naipopuliarnishi freimvorky dlia frontend rozrobky 2020 [Known and those hiding in the shadows: the most popular frameworks for frontend development 2020]. Retrived from: https://luxnet.io/uk/blog/popular-frameworks-for-front-end-development-2020 [in Ukrainian]
7. Frontend-2019: pidsumky roku [Front-End-2019: result of the year]. Retrived from: https://senior.ua/articles/frontend2019-pdsumki-roku [in Ukrainian]
8. React. Retrived from: https://uk.wikipedia.org/wiki/React [in Ukrainian]
9. Veb-zastosunok [Web-App]. Retrived from: https://uk.wikipedia.org/wiki/Veb-zastosunok [in Ukrainian]
10. Vue js. Retrived from: https://ru.wikipedia.org/wiki/Vue.js [in Ukrainian]