УДК 004.932.2

Кірша Е.В., магістрант, Попівщий В.І., доцент, науковий керівник,

ДОСЛІДЖЕННЯ І ПОРІВНЯННЯ ТЕХНОЛОГІЙ PROGRESSIVE WEB APP ТА NATIVE MOBILE APP

Запорізька державна інженерна академія, кафедра ПЗАС

Актуальність дослідження. Згідно з даними аналітичної компанії StatCounter, в листопаді 2016 року кількість підключень через мобільні пристрої вперше перевищила кількість підключень з використанням комп'ютерів і ноутбуків. Станом на липень 2017 року частка настільних комп'ютерів і ноутбуків знизилася до 43,3%, а частка мобільних пристроїв збільшилася до 56,7% від загальної кількості пристроїв в інтернеті [1].

Сьогодні 2,5 млрд жителів земної кулі користуються смартфонами, і 76% з них проводять перед екранами гаджетів більше трьох годин в день. Всі прогресивні компанії від банків до піцерій намагаються це використати, а успіх сервісів на зразок Uber і Airbnb зробив неймовірно популярною бізнес-модель, коли послугу можна замовити в застосунку, а потім отримати в офлайні. [2] В умовах значного зростання користувачів мобільними пристроями, росте і без того чималий попит на розробників застосунків.

В 2015 дизайнер Франсес Берріман та розробник Google Chrome Алекс Рассел створили термін "Progressive Web Apps" щоб описати застосунки, що використовують нові функції сучасних браузерів, такі як Service Workers та маніфести, і дозволяють користувачам встановлювати веб-застосунки, так як застосунки першого класу в своїй рідній ОС [3].

Беручи до уваги можливості сучасних веб-браузерів, є доцільним дослідити використання технологій Progressive Web App (PWA) для розробки мобільних застосунків.

Формулювання проблеми. Багато компаній стикаються з проблемою розробки різних застосунків для різних платформ. Їм часто доводиться розробляти два мобільних застосунки, один для iOS і один для Android. Крім того, їм необхідний веб-застосунок, який добре працює як на робочому столі, так і на мобільному пристрої. Однак знайти фахівців, у яких вже є досвід реалізації складних проектів, не так-то просто. Це відбивається на ринковій вартості їх послуг. Згідно зі статистикою Business of Apps, годинна ставка iOS- і Android-девелоперів в Східній Європі становить $35 [2].

Веб-застосунки часто описуються як крос-платформні. Вони доступні із великої кількості браузерів, котрі працюють на різних операційних системах. Виникає питання: чи здатні прогресивні веб-застосунки конкурувати, коли мова йде про час відгуку при доступі до апаратних засобів пристрою.

Мета роботи: на прикладі тестових застосунків порівняти продуктивність технологій Progressive Web App та Native Mobile App, а саме час відгуку при доступі до апаратних засобів мобільного пристрою.

Результати дослідження. Щоб дізнатися, які на даних момент функції надає браузер для доступу до апаратних засобів пристрою, використовувався сайт WhatCanTheWebDo.Today [5]. Цей ресурс надає список доступних API для конкретного браузеру. Для перевірки було обрано браузер Chrome, так як він має найбільш широку підтримку апаратних засобів.

Було розглянуто два сучасних фреймворка для розробки нативних мобільних застосунків, а саме React Native та NativeScript. Проаналізувавши ресурс stackshare.io [6] та інші ресурси, для подальшого дослідження було обрано фреймворк React Native. Вибір обумовлений його поширеністю та великою кількістю реалізованих проектів. Розробка нативного застосунку окремо під кожну платформу не розглядалась через ряд недоліків, серед яких висока ціна і довгий час розробки, складне і дорого технічне обслуговування, необхідність підтримки декількох версій застосунку.

Для розробки Progressive Web Apps було обрано React - відкриту JavaScript бібліотеку для створення інтерфейсів користувача, яка покликана вирішувати проблеми

часткового оновлення вмісту веб-сторінки, з якими стикаються в розробці односторінкових застосунків. React розробляється і підтримується такими великими компаніями, як Facebook та Instagram.

Для повноцінного PWA є необхідним використання Service Worker та маніфесту. Service Worker (Службовець) – це файл JavaScript, який працює у фоновому режимі. Він несе відповідальність за автономну функціональність прогресивного веб-застосунку за допомогою своєчасного кешування вмісту. Маніфест застосунку – це просто файл JSON. Він допомагає пристрою розпізнавати PWA і визначає, як він буде відображатися на екрані.

Час відгуку при доступі до апаратних засобів пристрою порівнювався завдяки використанню Performance API [7]. Для кожного порівняння було створено два застосунка: один для нативного мобільного і один для прогресивного веб-застосунку. У початковому представленні застосунку була поміщена кнопка. Коли кнопка натискається користувачем, застосунок звертається до пристрою. У той же час запускається тест. Тестування проводилися на Xiaomi Redmi 5 Plus (Android 8.1.0, MIUI Global 10.0.2) та LeTV Le One X600 (Android 5.0.2).

Висновки:

1. Були оглянуті, проаналізовані та систематизовані сучасні технології, фреймворки та інструменти для створення Native Mobile Apps та Progressive Web Apps.
2. Запропоновані характеристики порівняння застосунків.
3. Створене робоче середовища розробника, розроблені та протестовані застосунки.
4. Виконаний порівняльний аналіз різних технологій за запропонованими характеристиками.

Література

1. Desktop vs Mobile Market Share Worldwide [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: http://gs.statcounter.com/platform-market-share/desktop-mobile/worldwide/#monthly-200901-201707
2. Почему разработка мобильных приложений стоит дорого [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://livetyping.com/ru/blog/pochemu-razrabotka-mobilnyh-prilozhenij-stoit-dorogo
3. Progressive Web Apps [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Progressive_Web_Apps
4. Comparing Progressive Web Applications with Native Android Applications: An evaluation of performance when it comes to response time [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://www.diva-portal.org/smash/get/diva2:1105475/FULLTEXT01.pdf
5. What Web Can Do Today [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://whatwebcando.today
6. React Native vs. NativeScript [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://stackshare.io/stackups/nativescript-vs-react-native
7. Using the Performance API [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/API/Performance_API/Using_the_Performance_API