УДК 004.4'2 **doi: 10.26906/SUNZ.2020.1.074**

Н. В. Ічанська, С. І. Улько

Національний університет «Полтавська політехніка імені Юрія Кондратюка», Полтава,Україна

ОСНОВНІ АСПЕКТИ СТВОРЕННЯ МОБІЛЬНИХ ДОДАТКІВ ТА ВИБІР ІНСТРУМЕНТІВ ЇХ РОЗРОБКИ

Анотація. У роботі описано основні хронологічні аспекти розвитку мережі, наведено опис основних відмінностей нативної і багатоплатформної розробок мобільних додатків, подано їх порівняльний аналіз, аргументовано ефективність поєднання цих двох технологій. На думку авторів симбіоз цих розробок є універсальним та альтернативним варіантом для створення різноманітних проєктів, які, зазвичай, розробляються або нативно або багатоплатформно. Поєднання технологій розробок рекомендується авторами саме для тих проєктів, які не потребують високої продуктивності, але є достатньо масштабними, наприклад, навчальні додатки для освітніх закладів. Авторами подано оптимальний алгоритм відбору засобів створення мобільних додатків. Популярність мобільних платформ та мобільних додатків невпинно росте і тому ринок мобільних додатків має постійно оновлюватися і відповідати сучасним вимогам. Модернізація технологій розробки є **актуальною темою**, що викликає широкий інтерес замовників, розробників і користувачів. **Предметом** дослідження роботи є засоби реалізації створення мобільних додатків. **Мета роботи** – надати рекомендації по якісному вибору сучасних технологій та існуючих і широковживаних засобів розробки, що є найбільш популярними для створення мобільних додатків. **Результати** – проаналізовано сучасні технології та інструменти розробки мобільних додатків, наведено їх порівняльний аналіз. Аналізуючи проведене в роботі дослідження, користувач може створити власний мобільний додаток, наприклад, навчальний.

Ключові слова: фреймворк, мобільний додаток, нативний, багатоплатформний, мова програмування, JavaScript, React Native, Java, Kotlin, Objective C, Swift, Android, iOS.

Вступ

Ефективне існування кожної із сучасних галузей неможливе без застосування інформаційних технологій (ІТ). Усі сфери діяльності сучасної людини: спілкування, харчування, медицина, транспорт, послуги та інше використовують інтернет. Мережа стає невід'ємною та необхідною умовою успішного функціювання всіх ланок економічного простору. Кожен із нас є користувачем мережі й це є ознакою сучасного світу.

Аналіз проблеми та постановка задачі

Інтернет виник більше ніж півстоліття тому як результат рішення задачі комутації пакетів Агентства перспективних дослідницьких проєктів Міноборони США [1]. У 1969 році в Каліфорнійському університеті було вперше встановлено з'єднання, що дорівнювало відстані 640 км (Стенфордський університет), фінальною крапкою щодо розв'язання задачі комунікації став 1971 рік. Зауважимо, що саме у цьому році почала працювати перша мережева система електронної пошти [2].

Важливим етапом розвитку інтернету є 1984 рік, тоді стартувала перша комп'ютерна система розподілу доменних імен, яка дозволяла отримати інформацію про область сайту в мережі [1].

У 1988 році створення протоколу IRC (Internet Relay Chat) заклало основу першого чату, що забезпечило спілкування користувачів між собою в реальному часі [1].

Основним постачальником даних глобальна мережа стала лише у 1995 році, коли її трафік (за обсягом інформації, що пересилається) перевищив поширений протокол передачі файлів FTP (File Transfer Protocol) [1]. Це надало глобального значення термінам «інтернет» і «всесвітня мережа» та стало поштовхом для створення W3C (Консорціуму Всесвітньої павутини) [3].

Розвиток індустрії мобільних пристроїв спричинив спад актуальності використання персонального комп'ютера. Сучасний світ почав широко застосовувати мобільний інтернет і мобільні додатки. Функціональність мобільних додатків є надзвичайно різноманітною: від ігор та сервісів виклику таксі до офісних програм та фітнес-трекерів.

Із розвитком сучасних технологій зростає роль мобільних додатків, їх використання стає більш універсальним. Користувач хоче бачити використовувати телефон, роль персонального комп'ютера вже не така важлива. Компанії також надають більшу перевагу використанню мобільних додатків у порівнянні з сайтами. Для прикладу, у Tinder [4] до 2017 року не було веб-сервісу, а український Monobank [5] інтегрував цілий банк в один додаток, при цьому у Monobank існує сайт, але його роль в основному ознайомча, там подано посилання на скачування та умови отримання банківських послуг.

Невпинний рух вперед розвитку мобільних пристроїв породив ідею умонтування до телефону браузер. Це стало можливим завдяки WAP (Wireless Application Protocol) в 1998 році [6], результатом появи якого стало об'єднання інтернету та мобільного зв'язку. Але головним додатком у системі все ж залишався лише браузер.

У 2001 році операційна система (ОС) Symbian стала відкритою і в цей же час з'явилася Nokia 7650 з можливістю встановлювати додатки сторонніх розробників. Це був значний крок вперед та все ж смартфони ще мали обмежені можливості і викори-

стання мови C++ створювало певні складності у розробці [7].

У цей час відбувається стартовий розвиток ринку Java-додатків. Мова Java набуває популярності та часто використовується при розробці застосунків для Windows Mobile, Android, Bada, Palm OS і BlackBerry OS. Зауважимо, що у операційній системі Symbian, на той момент, існувала можливість підтримки підмножини Java – J2ME, але функціональність таких програм була дещо обмежена [8].

У 2007 році Стів Джобс представив світу перший iPhone, де було подано достатньо повний перелік додатків. Архітектура iOS була схожа на MacOS, але система була повністю закритою. С. Джобс не хотів надавати доступ стороннім розробникам до програм для iOS, а отже, не збирався відкривати SDK (Software Development Kit) [9]. Хоча при цьому С. Джобс сприяв подальшому розвитку індустрії веб-додатків для iPhone і тому він дав можливість створювати браузерні закладки на домашньому екрані. Але допитливі зламали файлову систему, почали писати інсталятори для додатків і заодно – самі додатки. Так з'явився джейлбрейк (відкриття файлової системи iOS-пристроїв). Пізніше рада директорів Apple все ж переконав Джобса легалізувати сторонні додатки. У підсумку в березні 2008 року набір засобів розробки iPhone SDK стає доступним всім бажаючим, а в липні світ отримав магазин додатків App Store. Це означало, що Apple організовує дистрибуцію продуктів розроблених користувачами. App Store став поштовхом до розвитку індустрії розробки додатків, та все ж мова програмування Objective-C залишалася проблемною [9].

Objective-C для iOS кардинально відрізнялася від популярних тоді скриптових мов JavaScript і Flash Action Script. У той час зростає популярність мобільних ігор, що призводить до збільшення попиту на їх розробку, а значить і збільшення прибутків від них. Та все ж, попит на розробку сервісів і додатків для бізнесу ще не є достатньо великим. Тільки з появою у 2008 році Android Market, а в 2010 році – Windows Mobile Store різноманітність додатків суттєво збільшилася. Ці дистрибуційні платформи дали поштовх для розробників не тільки ігрових додатків [8].

Розвиток мобільного інтернету набирає високих обертів і разом з цим розширюється набір інструментів SDK, різноманітні платформи отримують цікаві рішення з безпеки та інтеграції. Це був новий етап розвитку в розробці додатків: від розваг розробники перейшли до вирішення завдань бізнесу.

Швидкий паралельний розвиток iOS і Android створив двополярну систему, що породжує перед розробниками нову задачу, а саме: підтримання кілька платформ одночасно. На той момент з багатоплатформних інструментів використовувалися лише Flash і звичайний мобільний браузер, причому в 2010 році Apple відмовилися від підтримки технології Adobe Flash в iOS. Розв'язання цієї задачі, навіть в простому рішенні для різних платформ впиралася в людські, часові та матеріальні ресурси. На допомогу прийшли бібліотеки компонентів і фреймворки Xamarin, Cordova, Phonegap для створення додатків на Android і iOS на базі браузерних технологій без використання мов програмування.

Фреймворки – це програмні продукти, які спрощують створення і підтримку технічно складних або навантажених проєктів [10]. Фреймворк, як правило, містить тільки базові програмні модулі, а всі специфічні для проєкту компоненти реалізуються розробником на їх основі. Що надає не тільки високу швидкість розробці, а й гарантує велику продуктивність і надійність рішень. За допомогою фреймворків створюється мобільний сайт, зверху накладається платформний код, який транслює двосторонні виклики між системою та додатком. З появою перших фреймворків (Xamarin, Cordova, Phonegap) було розв'язано деякі проблеми маленьких додатків з невеликим функціоналом. Та все ж, проблеми продуктивності, споживання ресурсів, чутливості багатоплатформних додатків і «чужорідного» дизайну залишилися відкритими [8].

У 2015 році розробниками Facebook на конференції React.js Conf було представлено свій інструмент для багатоплатформних рішень – фреймворк React Native [11], де компоненти програми, написані на JS, транслюються в нативні Android і iOS. Цей інструмент принципово відрізняється від інших систем для створення багатоплатформних додатків:

– відсутністю WebView і HTML-технологій;

– візуалізацією інтерфейсу. У RN її виконує ОС пристрою, а не браузер;

– відсутністю додаткової «обгортки» коду – замість неї JS взаємодіє з ОС через спеціальний міст.

Завдяки цим відмінностям додатки з використанням React Native максимально схожі на нативну розробку та менш конфліктні з продуктивністю.

У цей час з'являються мови програмування Swift і Kotlin як рішення задачі мінімізації складнощів при розробці нативних додатків. Swift було представлено компанією Apple на конференції WWDC (Worldwide Developers Conference) в 2014 році. Ця мова має багато спільного з Objective-C, але вона працює за аналогією зі скриптовими мовами. Її код визначається типами змінних, а не покажчиками. Це робить вивчення Swift простішим для тих, хто вже володіє будь-якою скриптовою мовою.

Kotlin з 2010 року розробляла компанія JetBrains [12] з метою створення більш лаконічної і простої мови в порівнянні з Java, яка вже накопичувала багаж невдалих рішень. З 2017 Kotlin офіційно рекомендується Майком Клероном – одним з директорів Google для Android-додатків.

Сьогодні, 99,9% смартфонів функціонують на операційній системі iOS або Android [13]. Більшість компаній використовують мобільні додатки як канал інформування користувачів, а не як спосіб ведення бізнесу. Економічний розвиток компаній, їх стабільність та успішне існування підкреслює

актуальність створення нативних або багатоплатформних розробок і в освітній галузі також. Розробка додатку дає можливість навчальному закладу покращити якість освіти та набагато ширше презентувати свій навчальний потенціал: електронні бібліотеки, книги, додатки для спрощення навчання чи тестування.

У цій роботі подано опис основних відмінностей нативної та багатоплатформної розробок мобільних додатків, проаналізовано їх переваги та недоліки.

Тема вибору правильних інструментів для створення мобільних додатків викликає інтерес багатьох розробників [14-18]. Переваги нативної чи багатоплатформної систем серед розробників породжують активну дискусію.

Автори цієї роботи вважають, що доцільно використовувати поєднання обох технологій. Цей симбіоз вперше було запропоновано Марком Цукербергом – президентом і генеральним директором Facebook та Джорданом Волком – інженером програмного забезпечення Facebook у 2015 році як фреймворк React Native [19]. До сьогоднішнього дня він знаходиться ще на стадії розробки, але вже встиг завоювати довіру багатьох розробників мобільного програмного забезпечення. Рекомендована технологія розробки є ефективною для саме тих проєктів, які не потребують високої продуктивності, але можуть бути масштабними, зокрема, додатки для навчальних закладів.

У роботі розглянуто алгоритм відбору засобів створення мобільного додатку. Предметом дослідження є засоби реалізації створення мобільного додатку.

Мета роботи – надати рекомендації по вибору інструментів для розробки додатку навчального закладу.

Виклад основного матеріалу

Розглянемо та порівняємо основні інструменти для розробки додатків, наприклад, навчальних. Мобільні додатки можна поділити на дві категорії: нативні та багатоплатформні.

Нативні додатки [19] – це додатки, що розроблено для конкретної платформи (iOS чи Android) з урахуванням її специфіки та доступом до всіх ресурсів.

Багатоплатформні додатки [19] – це універсальні додатки, що створено для кількох платформ одразу і мають аналогічну функціональність незалежно від самої платформи.

Зазвичай це веб-сайт у звичному форматі мобільного додатку.

Розробка нативного додатку – це технологія, яка відповідає вимогам ядра операційної системи з використанням SDK (а також з наданням можливості роботи з пам'яттю, внутрішніми датчиками пристрої та іншими встановленими додатками) [19].

Переваги нативних додатків включають в себе:

– високу продуктивність;
– основний призначений для користувача досвід;
– велику видимість в магазинах додатків.

За допомогою "рідних" кожній платформі інструментів створюється величезна кількість додатків, але вони не є ідеальні.

Недоліки нативного додатку полягають в тому, що в його розробці будуть задіяні декілька команд, в неможливості перевикористання коду, як наслідок – більша витрата часу.

Розробка багатоплатформного додатку – це технологія, яка сумісна з безліччю операційних систем і таким чином працює на різних смартфонах і планшетах [20].

Існує два типи реалізації багатоплатформної розробки: нативні багатоплатформні та гібридні. Опишемо кожен з них.

Нативні багатоплатформні додатки. Кожна операційна система має власний SDK і технічний стек (Java для Android і Objective-C / Swift для iOS). Досвідчений розробник може створити єдиний API, що функціонує на native Software Development Kit і використовує одну і ту ж кодову базу для Android і IOS. Нативні багатоплатформні програми зазвичай створюються за допомогою Xamarin, React Native, NativeScript.

Гібридний HTML5 додаток. Хоча, мобільні додатки розробляються для смартфонів і планшетів, їх бекенд сервери справляються з логікою програми. З тих пір як IOS і Android SDK почали характеризуватися як просунуті веб-компоненти, кваліфіковані розробники програмного забезпечення часто використовують WebView, щоб створити різні частини програми GUI (Graphical User Interface) з використанням HTML5, CSS і JavaScript.

Найбільш популярним засобом є фреймворк – Apache Cordova (раніше відомий як PhoneGap).

Опишемо основні інструменти багатоплатформних розробок.

Apache Cordova. Технічний стек Apache Cordova складається з HTML5, CSS3 і JavaScript. Механізми розробки мобільних додатків дають доступ до вбудованого акселерометру смартфону, сховища файлів, GPS, контактним даним, мультимедіа та оповіщенням. Apache Cordova також може похвалитися кількома перевагами включаючи досить простий API і можливістю використовувати будь-який JS фреймворк. Проте, платформа являє UI (User Interface) інтерфейс програми через веббраузер (який, звичайно, може зависати при великих додатках).

Xamarin. Додатки на Xamarin створюються за допомогою C# і .Net. Xamarin дозволяє розробникам повторно використовувати код і полегшує процес створення динамічного макета для IOS. Однак компоненти UI не можуть бути здійснені на MonoTouch і MonoDroid, з тих пір як вони поклалися на специфічні особливості Android і IOS.

React Native. Рішення від Facebook, яке є логічним продовженням React, але для мобільної розробки та використовує:

– мову програмування JavaScript;
– нативний UI для кожної платформи;
– UI як функцію поточного стану.

UI виділяється в окремі компоненти для забезпечення промальовування стану.

Дуже важливо правильно організувати data-flow (потік даних), а також налагодити взаємодію між компонентами. Це необхідно для хорошої організації структури коду.

Ізольованість UI дає можливість великого перевикористання коду між Mobile- і web-рішенням.

Native Script. Фреймворк використовує:

– нативний рендеринг без WebView;
– фреймворк Angular 2;
– стандартний двосторонній біндинг.

Зауважимо, що NativeScript у порівнянні з:

– Phonegap не потребує жодних плагінів для використання нативних API;
– ReactNative використовує Angular2.

Опишемо основні переваги та недоліки багатоплатформних додатків.

Перевагами розробки багатоплатформної програми є:

1. Більш короткий час розробки. Якщо обрати правильний технічний стек і розпланувати ретельно проєкт, можна перевикористати до 80% коду.

2. Рентабельність. Розробка нативного мобільного додатку є більш вартісна у порівнянні з розробкою багатоплатформного мобільного додатку.

3. Вплив на велику кількість користувачів. Багато багатоплатформних додатків працюють на IOS і Android системах (також на Windows, Linux, Tizen і навіть Symbian).

4. Синхронізація оновлень. У світі, де розробники додатків впроваджують оновлення приблизно 4 рази на місяць, технічне обслуговування забирає більшу частину прибутків, що підкреслює виграшність багатоплатформних розробок.

Недоліками розробки багатоплатформної програми є:

1. Проблеми з продуктивністю. Обчислювальна потужність смартфонів відносно мала. Сприяння матеріально-технічного забезпечення HTML5 / CSS UI, навпаки, вимагає багато ресурсів GPU (Graphics Processing Unit) / CPU (Central Processing Unit) і може збільшити час відгуку програми.

2. Проблеми з дизайном. Поєднання UX (User Experience) вимог до дизайну двох платформ може викликати складнощі. Apple відомі своїм Human Interface Guidelines і відхиляють веб-сайти в нативній обгортці. Однак 20% відмов у App Store припадають на баги (помилки в програмі) і поганий UI дизайн. Та цю проблему легко може усунути надійна компанія з розробки гібридних багатоплатформних додатків.

Вибір між нативною і багатоплатформною розробкою програми залежить від набору функцій програми та її призначення.

Якщо передбачається, що додаток стане самостійним бізнесом (наприклад, фітнес додатки, додатки по турботі про здоров'я), то слід обирати нативні способи розробки. Для компаній, які замовляють суто інформативні додатки (або додатки для поширення інформації), краще використовувати багатоплатформні розробки.

Зауважимо, що вартість проєкту суттєво зростає, при розробці повністю нативного додатку на Android та iOS окремо. Тоді є два варіанти розв'язання цієї проблеми замовити або гібридний, або нативний багатоплатформний додаток. При цьому, у першому випадку, витрати на розробку є мінімальними, проте додаток може бути ненадійним (його легко зламати) та повільним у роботі. Щодо другого випадку, вартість більша у порівнянні з витратами на створення гібридного додатку, але значно менша за вартість повністю нативної розробки. Отриманий таким чином додаток є продуктивним та достатньо безпечним, оскільки вся його логіка скомпілюється у повністю бінарний код.

Існує багато різних платформ, та все ж основним є платформи: Android та iOS. Інші з часом втратили свою актуальність і тому засоби розробки мобільних додатків мають базуватися на цих двох платформах. Це є продуктивно і не звужує коло користувачів.

Одним із важливих критеріїв популярності мобільних додатків є ергономіка. Ергономіка застосунку – це зручність користування. Запорука ергономічності будь-якої мобільної розробки полягає у використанні звичного для платформи дизайну, нативних елементів та логічного управління. На зручність додатку впливає відповідність його дизайну тій платформі під яку він створюється. Але при створенні гібридного додатку 100% «схожості» із платформою досягти важко, оскільки HTML ніяк не зможе замінити нативні компоненти. Інколи розв'язання цієї проблеми можливе, але це призводить до збільшення розміру додатку та зниження його швидкодії.

При розробці нативного додатку цієї проблеми не виникає. При створенні нативного багатоплатформного додатку стандартні компоненти вже присутні в фреймворці, а дизайн допускає налаштування відповідні до конкретної платформи.

Проведений порівняльний аналіз технологій розробки, показує що:

1. Розробка гібридного додатку є доцільною при організації невеликих за масштабом проєктів, які не потребують складної логіки та дизайну. Наприклад, інформаційний портал.

2. Розробка нативного додатку є доцільною при організації швидкодійних, безпечних та багатоструктурних за своїм дизайном проєктів. Наприклад, соціальна мережа чи додаток для банку.

3. Розробка нативного багатоплатформного додатку є найбільш універсальною. Реалізує основні задачі масштабних проєктів, гарантує швидкодію, безпеку та дизайн відповідно базової платформи. Наприклад, додаток навчального закладу з власним форумом та можливістю подавати новини, проводити онлайн тестування чи опитування.

Висновки

Поданий у роботі аналіз інструментів та засобів можна практично реалізувати в процесі створення мобільного додатку, а саме при виборі інструментів розробки. Рекомендації, що подані в роботі, дають

можливість швидко та ефективно створювати мобільні застосунки, що володіють швидкодійністю та універсальністю, є дешевими в розробці, простими в підтримці, нескладними в обслуговуванні. Правильний аналіз розглянутих методів та засобів дозволить обрати правильну розробку конкурентоспроможного мобільного додатку, яка буде мінімальною по часовим та матеріальним витратам, але водночас дасть безпечний, інтуїтивно зрозумілий та швидкодіючий додаток.

Список літератури

1. Історія виникнення інтернету, 2016, [Електронний ресурс] – Режим доступу до ресурсу: https://webbuilding.com.ua/ukr/articles/istoriya-interneta/
2. A brief history of email: dedicated to Ray Tomlinson, 2016, [Електронний ресурс] – Режим доступу до ресурсу: https://phrasee.co/a-brief-history-of-email/
3. W3C [Електронний ресурс]. Доступно: https://www.w3.org/
4. Tinder, [Електронний ресурс] – Режим доступу до ресурсу: https://tinder.com/?lang=uk
5. Monobank | Universal Bank, [Електронний ресурс] – Режим доступу до ресурсу: https://www.monobank.ua/?lang=uk
6. Wireless Application Protocol 2.0, 2001, [Електронний ресурс] – Режим доступу до ресурсу: http://www.informit.com/articles/article.aspx?p=23999
7. Восход и закат Symbian, 2011, [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/128361/
8. Эволюция в вашем кармане: как развивались мобильные приложения, 2018, [Електронний ресурс] – Режим доступу до ресурсу: https://apptractor.ru/info/articles/evolyutsiya-v-vashem-karmane-kak-razvivalis-mobilnyie-prilozheniya.html
9. Walter Isaacson "Steve Jobs", Simon & Schuster, 2011
10. Framework, [Електронний ресурс] – Режим доступу до ресурсу: https://whatis.techtarget.com/definition/framework
11. React Native. Build native mobile apps using JavaScript and React, [Електронний ресурс] – Режим доступу до ресурсу: https://facebook.github.io/react-native/
12. Kotlin, [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Kotlin
13. iOS и Android занимают уже 99,9% рынка мобильных ОС, 2018, [Електронний ресурс] – Режим доступу до ресурсу: https://www.ixbt.com/news/2018/02/24/iOS-android-99-9.html
14. Native vs. Cross-Platform Apps – You'll Be the Winner, 2018, [Електронний ресурс] – Режим доступу до ресурсу: https://www.zeolearn.com/magazine/native-vs-cross-platform-apps-youll-be-the-winner
15. Cross-platform vs. Native Mobile App Development: Choosing the Right Dev Tools for Your App Project, 2017, [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/all-technology-feeds/cross-platform-vs-native-mobile-app-development-choosing-the-right-dev-tools-for-your-app-project-47d0abafee81
16. Cross-platform vs native app development?, 2019, [Електронний ресурс] – Режим доступу до ресурсу: https://www.brightec.co.uk/ideas/cross-platform-vs-native-app-development
17. Омелян, О. М.; Ічанська, Н. В. "Використання інформаційно- комунікаційних технологій у процесі викладання математики", Математика в сучасному технічному університеті: Матеріали Шостої міжнародної науково-практичної конференції, (Київ 28-29 грудня, 2017 р.) – Київ: КПІ імені Ігоря Сікорського (Київ), 2018. – С. 370-374.
18. Методологія підвищення якості функціонування інформаційно-телекомунікаційних систем: Монографія / О. В. Шефер. – Полтава: ПолтНТУ імені Юрія Кондратюка, 2019. – 236 с.
19. React Native, [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/React_Native
20. Типи мобільних додатків, [Електронний ресурс] – Режим доступу до ресурсу: https://smile-ukraine.com/ua/mobile-apps/mobile-apps-types

Received (Надійшла) 26.11.2019
Accepted for publication (Прийнята до друку) 22.01.2020

Main aspects of creating mobile applications and selecting the instruments for their development

N. Ichanska, S. Ulko

Abstract. The paper describes the main chronological aspects of network development, describes the main differences between native and multi-platform development of mobile applications, presents their comparative analysis, shows the effectiveness of combining these two technologies. According to the authors, the symbiosis of these developments is a universal and alternative option for creating various projects that are usually developed either native or multi-platform. The combination of development technologies is recommended by the authors specifically for those projects that do not require high productivity, but are large enough, for example, educational applications for educational institutions. The authors presented the optimal selection algorithm for creating mobile applications. The popularity of mobile platforms and mobile applications is constantly growing, and therefore the mobile application market must be constantly updated and meet modern requirements. Modernization of development technologies is an **urgent** topic, which causes widespread interest of customers, developers and users. The **subject** of research is the means of implementing the creation of mobile applications. The **purpose** of the work is to give recommendations on the high-quality selection of modern technologies and existing widely used development tools, which are the most popular for creating mobile applications. **Results -** modern technologies and tools for developing mobile applications are analyzed, their comparative analysis is given. Analyzing the research carried out in the work, the user can create his own mobile application, for example, a training one.

Keywords: framework, mobile application, native, multi-platform, programming language, JavaScript, React Native, Java, Kotlin, Objective C, Swift, Android, iOS.