ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА АКТУАЛЬНИХ ЗАСОБІВ РОЗРОБКИ ПІД МОБІЛЬНІ ПЛАТФОРМИ

УДК 004.9

ЯШИНА Оксана Миколаївна
к.т.н., доцент кафедри ІПЗ Хмельницького національного університету
Наукові інтереси: дослідження та впровадження смарт-технологій у різні сфери життєдіяльності людини.
e-mail: ksusha.ja@gmail.com.

ГРЕМЕЧЕВСЬКИЙ Руслан Васильович
студент Хмельницького національного університету
Наукові інтереси: дослідження та прикладне застосування останніх напрацювань програмної інженерії у сфері проектування та розробки мобільних додатків.
e-mail: grem.rusl@gmail.com.

БРАТАСЮК Денис Іванович
студент Хмельницького національного університету
Наукові інтереси: дослідження та впровадження смарт-технологій у різні сфери життєдіяльності людини.
e-mail: mqaak1@gmail.com.

ВСТУП

Із бурхливим розвитком інформаційних технологій, що дозволяють створювати різні мобільні пристрої, ринок програмних продуктів отримує потужний стимул до розвитку та вдосконалення. Сучасну мобільну техніку люди носять із собою завжди та всюди (у світі нараховується близько п'яти мільярдів користувачів смартфонів та інших мобільних гаджетів). Смартфони, планшети та інші пристрої відіграють значну роль у побуті, особистому житті людей, роботі (є можливість легко та швидко прочитати файл, зайти на пошту, надрукувати документ за допомогою мережевого принтера тощо). Із збільшенням продажу мобільних пристроїв поступово сформувався окремий напрямок програмного забезпечення – мобільні додатки. Інформаційні технології вливаються у бізнес-процеси сучасних підприємств все інтенсивніше, стаючи більш зручними у використанні, доступними по ціні та легкими у використанні. Важко назвати бізнес, що не використовує сучасні інформаційні технології. Однак, одночасно із таким бурхливим розвитком постає питання про зручність отримання даних. Саме мобільні технології здатні прийти на допомогу електронному бізнесу. Мобільні технології – остання та найбільш актуальна тенденція розвитку ринку інформаційних технологій для автоматизації бізнес-процесів. Смартфони та планшети все більше проникають у корпоративне середовище і перетворюються для багатьох керівників та співробітників із особистого мобільного пристрою у робочий інструмент. Керівники компанії та організацій задумуються про те, щоб централізовано впроваджувати та використовувати ці технології в інтересах бізнесу. Розробка програм для мобільних пристроїв у сфері бізнесу необхідна для просування товарів чи послуг широкій аудиторії. Якісний та зручний додаток користувачі будуть активно використовувати та рекомендувати друзям та знайомим. Якщо створювати додаток із врахуванням інтересів потенційних користувачів, то це підвищить лояльність та довіру до бренду, а також допоможе оптимізувати виконання вирішуваних задач. Так, наприклад, мобільний банк дає можливість контролювати, оплачувати рахунки, здійснювати переведення коштів тощо.

Розробка мобільних додатків актуальна у багатьох сферах, таких як транспортні послуги, мережі ресторанів та кафе, інтернет-магазини, журнальний бізнес, організація корпоративного спілкування і багатьох інших підприємств та торговлі продукцією.

В сучасних умовах підхід Mobile First залучається у всі сектори економіки, замінивши попередній робочий процес, починаючи з веб-сайту та закінчуючи задоволенням потреб користувача на кінцевому шляху. Зі зростанням популярності мобільної платформи, було впроваджено багато технологій, що полегшують просування у сфері смартфонів.

Це почалося з гібридних фреймворків для розробки додатків, що дозволяють використовувати основні веб-технології, такі як HTML, CSS та JavaScript для компіляції та розгортання додатків на різних платформах. Потім з'явилися нативні фреймоворки, що не тільки дозволяють використовувати веб-інструменти для мобільної розробки, але також використовують нативний API для Android чи iOS.

Ці платформи служать такій же меті, як підтримка та створення мобільних додатків за допомогою веб-технологій, таких як HTML, CSS, JavaScript. Те, чим вони відрізняються, - це функції, що вони надають, і як вони спілкуються з API для конкретних платформ.

Метою статті є визначення характеристик сучасних засобів розробки мобільних додатків, обґрунтування доцільності використання кожного із них в залежності від вимог до кінцевого продукту.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

Зазвичай під мобільним бізнесом розуміють традиційні послуги електронного бізнесу, який забезпечується бездротовими мережами. Однак, мобільний бізнес забезпечує і принципово нові сервіси, які не можуть бути реалізовані за допомогою звичайної мережі, тому він виходить за її межі .

Отже, актуальність розробки мобільного додатку для керування бізнес-процесами на підприємстві, організації чи установі не викликає сумнівів..

Не зважаючи на стрімкий розвиток мобільних платформ та, відповідно, зростаючий попит на методи та засоби стосовно розробки для них, об'єми наукової інформації у цій сфері все ще невеликі та, у більшості випадків, неактуальні. Звідси й підвищений науковий інтерес до даної теми.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Розглядаючи спосіб створення додатків iOS та Android, укорінюється думка, що єдиним вибором є споріднені мови Objective-C, Swift та Java. Однак цей підхід має серйозний недолік: при розробці відразу на дві платформи є необхідність двічі писати ті ж фрагменти коду. Розглядаючи різні мови програмування (Java і Swift), різні підходи до того, щоб робити те ж саме в iOS та Android, різні способи створення графічного інтерфейсу, важко зробити що-небудь, щоб двічі не писати той самий код. Ситуація ще гірша, якщо два різних розробники створюють програми для iOS і Android, оскільки дві людини, ймовірно, реалізують однакову функцію трохи по-різному. Проте за останні кілька років виникла ціла нова екосистема платформ для створення мобільних додатків.

Немає жодних сумнівів стосовно того, що коли мова йде про розробку мобільних додатків - лідерами швидкодії є продукти, написані нативними мовами [1,2,7]. Це мінімізує рівні абстракцій. Принцип «менше обміну даними - краща швидкодія» як ніде краще працює на мобільних платформах, де системні ресурси є сильно обмеженими. Проте, подекуди різниця у бюджеті та терміни розробки набагато серйозніші за різницю у швидкодії. Вихід на ринок таких гравців як React Native чи Xamarin зробили цю прірву ще більш значущою, тому питання стало гострішим.

Ionic, Cordova і PhoneGap - засоби для розробки гібридних додатків. Найбільша проблема з гібридними додатками - це їх швидкодія та ефективність. Спочатку веб-сайти створювались для простих веб-сторінок, а не складних додатків, які розробляються наразі. Гібридні програми працюватимуть без перебої на високотехнологічних пристроях, але все ще не так гладко, як хотілося б. Обмеженість системних параметрів телефонів є найбільшою проблемою.

Зазначені засоби дають змогу програмістам створювати програми для мобільних пристроїв за допомогою CSS3, HTML5 та JavaScript, замість того, щоб спиратися на API, специфічні для платформи Android, iOS або Windows Phone. Це дозволяє загортати код CSS, HTML та JavaScript в додаток, залежно від платформи пристрою.

Такі засоби розширюють можливості HTML та JavaScript для роботи з пристроєм. Проте отримані додатки будуть гібридними, а це означає, що вони не є нативними мобільними додатками (оскільки рендер представлення здійснюється за допомогою WebView замість власних елементів інтерфейсу платформи). Не можна їх називати і просто веб-додатками (оскільки вони упаковані як додатки конкретної платформи для розповсюдження та доступу до API пристроїв).

Інша проблема полягає в тому, що при розробці гібридного додатку автоматично успадковуються всі проблеми, які має розробник веб-сайтів. Як сайти можуть працювати по різному на деяких браузерах - так і додаток працюватиму зі своїми нюансами на певних пристроях. Діагностування помилок та відлагодження стає просто нестерпним для розробника.

Статистика говорить про те, що власники смартфонів витрачають більшу частину свого часу, використовуючи лише кілька додатків, і вони очікують, що будь-який новий додаток, який вони беруть у користування, буде швидким як Facebook, YouTube чи Uber. З високими сподіваннями користувачів гібридні додатки не справляються, пропонуючи більш обмежені можливості для користувачів з повільними анімаціями, відсутністю розпізнавання жестів, подекуди неправильною поведінкою клавіатури чи недостатньою взаємодією із мобільною платформою та її операційною системою.

Всі вищевказані проблеми ведуть до того що популярність гібридних додатків знижується. PhoneGap і Ionic демонструють помітні недоліки у великих додатках. На відміну від цього, список додатків, які перейшли з гібриду до рідного інтерфейсу на React Native стрімко збільшується. До списку входять Facebook, Airbnb, Instagram, Skype, Tesla, Slack та багато інших.

Найбільша користь, яку пропонує React Native, - це набагато краща продуктивність, ніж у випадку гібридних додатків. Розробник отримує доступ до всієї платформи і всіх нативних елементів інтерфейсу. Це дозволяє створювати додаток зі складними анімаціями та великою кількістю елементів управління. Крім того, апаратні функції обробляються конкретною платформою. React Native працює без використання WebView, тому розробнику не потрібно турбуватися про проблеми із сумісністю браузера. Нарешті, React Native базується на React структурі з набагато кращою підтримкою - як від Facebook, так і від активної та чисельної спільноти. Завдяки постійній підтримці, React Native залишається актуальним, забезпечує більш високу надійність та чудову базу знань, де можна дізнатись як розробляти додатки та вирішувати проблеми.

Порівнюючи засоби розробки під мобільні платформи було б неправильно не пригадати ще один. Xamarin - це один з найстаріших крос-платформних фреймворків такого типу. Проект був придбаний компанією Microsoft у 2016 році та став частиною їх IDE Visual Studio, що забезпечило платформі стабільний подальший розвиток. Це одна з ключових причин, чому великі компанії, такі як, наприклад, Pinterest, покладаються на Xamarin. Він пропонує єдину мову - C#, бібліотеку класів і runtime, що працює на трьох мобільних платформах: iOS, Android та Windows Phone (рідна мова Windows Phone і є C#). Швидкодія та продуктивність продукту на виході достатні навіть для вимогливих ігор.

Все ж вибір конкретного патерну залишається за розробником, який визначає більш кращий шаблон для конкретного випадку. Мобільні технології в бізнесі є логічним продовженням електронного документообігу і часто розуміється як його підмножина.

Стосовно популярності розглянутих засобів було зроблено вибірку статистичних даних сервісу Google Trands (табл.1).

Таблиця 1

Вибірка статистичних даних сервісу Google Trands

	React Native	Xamarin	Ionic	Cordova	PhoneGap
В Україні	40%	25%	24%	10%	1%
У світі	18%	14%	44%	22%	2%

Статистика показує що React Native лідирує в Україні, Ізраїлі, Швеції та деяких інших країнах. Проте у світі більш популярним запитом в Google є Ionic. Загальна тенденція показує що React Native та Xamarin з часом впевнено відбирають популярність у технологій що працюють на WebView.

Так чи інакше, якщо порівнювати нативні та гібридні додатки то постає питання про перевагу того чи іншого виду у 2018 році. Для цього потрібно з'ясувати скільки додатків використовує звичайний користувач щодня

та скільки додатків він не відкривав з моменту встановлення.

Загалом же завантаження додатка займає кілька хвилин, а також кілька секунд, щоб видалити його, якщо він не відповідає вашим очікуванням. Користувацький інтерфейс дуже важливий для мобільних додатків: привабливі елементи інтерфейсу користувача, плавна прокрутка, розпізнавання жестів для платформи, розширені анімації та ефекти. На відміну від цього, гібридна розробка додатків повинна адаптуватися до специфіки користувацького інтерфейсу декількох платформ одночасно. Незважаючи на те, що кросплатформенні технології та системи активно вдосконалюються, гібридні додатки все ще можуть виглядати та некоректно працювати на різних пристроях. Таким чином, якщо є зацікавленість в користувацькому інтерфейсі, варто вибрати нативну розробку додатків. Щоб захистити гібридний підхід, можна сказати, що хороший веб-дизайнер може подолати недоліки UX і зробити додатки дуже схожими до нативного дизайну. Але є й випадки, такі як Google Gmail, коли розробники не приділяють великої уваги рідному користувальницькому інтерфейсу та реалізують один дизайн для всіх платформ. Зрештою, користувачі не дуже піклуються про особливості вигляду і навіть не можуть заперечувати, що перед ними не нативний додаток (вони навряд чи думають про такі речі), якщо він відповідає всім своїм цілям.

Нативні додатки безпосередньо зв'язуються з апаратним забезпеченням пристрою, і, як правило, весь статичний вміст завантажується після встановлення додатка [2]. Таким чином, користувачі не залежать від швидкості інтернету, тому програма завжди плавно і надійно працює. На відміну від нативних додатків, гібридним спочатку потрібно пройти гібридну платформу і виконати код, а потім звернутися до пристрою. Як правило, вміст завантажується з сервера (хоча і не завжди), а продуктивність залежить від швидкості вашого інтернет-з'єднання. Крім того, можна відмітити деяке відставання, наприклад, переключивши IPad з вертикального в горизонтальне положення. Це є великим мінусом гібридного додатка. У нативному додатку зміна орієнтації екрана є швидкою та легкою, на відміну від гібридного (Facebook став швидшим з моменту перехоу на нативний додаток).

Однак, варто відмітити, що гібридні програми все ще повільніші, ніж їх конкуренти, але не так критично. Продуктивність - це індивідуальна річ, яка залежить від потреб проекту та можливостей пристрою. Розробнику та кінцевому користувачу потрібно звертати увагу на швидкість, якщо метою є створення додатук, що підтримує оновлення в режимі реального часу, високі навантаження (соціальні мережі, такі як Facebook), складну бізнес-логіку з кількома розрахунками (фото, музика або відеоредактори) та 3D-графіку (ігри). В інших випадках можна не звертати на це уваги. Наприклад, власник, може думати, що "прокрутка" на веб-сайті працює з деякими відставаннями, а користувачі навіть не звертають уваги на це [3]. Загалом же ефективність гібридних додатків, у останні роки, зросла завдяки наступним факторам:

1. Щороку пристрої стають набагато потужнішими та швидшими;
2. Дія ОС стає кращою від версії до версії;
3. Прогрес технологій;
4. Рівень кваліфікації та досвід розробників зростає.

Якщо говорять про використанні гібридних додатків з низькою продуктивністю, то це стосується старих пристроїв, застарілих ОС, недосвідчених розробників або поганої оптимізації використовуваних технологій.

Гібридні програми можуть перемогти свого конкурента лише в швидкості розробки програмного продукту. Маючи єдину кодову базу, ці додатки розробляються швидше, тому власник чи розробник може представити додаток на ринку вже через кілька місяців. Для пристосування функцій до різних платформ у гібридному підході потрібно менше часу, ніж створення дизайну для всіх можливих параметрів пристроїв (iPhone, планшетів, різних смартфонів Android тощо) у нативній розробці додатків. До речі, зміни в гібридах швидше доставляють користувачам, не заставляючи їх завантажувати оновлення.

Зазвичай розробники спеціалізуються лише на одній платформі (iOS або Android). Отже, щоб створити нативний додаток для різних платформ, знадобиться щонайменше дві команди розробників. Це також означає два бюджети. Будь-які подальші зміни та оновлення також дадуть кошти, помножені на кількість платформ. Розробка гібридних додатків є дешевшою, оскі-

льки деякі рішення швидше створюються за допомогою HTML та JS, аніж Objective-C або Java. Крім того, більша частина коду може бути повторно використана для різних платформ, хоча все ще існує необхідність адаптуватися до специфіки платформ. У будь-якому випадку знадобиться менше ресурсів і менше бюджету.

Далі подано переваги та недоліки нативних та гібридних.додатків [4].

Переваги нативного додатку:

1. Набір багатих нативних інтерфейсів, який добре знайомий користувачам і допомагає їм інтуїтивно вивчати додатки.

2. Повний потенціал платформи (багато готових та перевірених рішень).

3. Висока швидість виконання.

4. Прямий доступ до функцій пристрою (камера, GPS, адресна книга, календар, push-сповіщення) через API.

Недоліки нативного додатку:

1. Тривалий розвиток, оскільки вимагає знання в різних мовах програмування.

2. Набагато дорожче, оскільки вам потрібно нагодувати більше фахівців для створення та підтримки різних платформ.

Переваги гібридного додатку:

1. Швидка розробка: одна кодова база для всіх платформ

2. Просте тестування та впровадження багатьох змін та масштабування на інші платформи.

3. Дешевше, адже оплата здійснюється лише одній команді.

Недоліки гібридного додатку:

1. Не такий гладкий і інтуїтивно зрозумілий UX як у нативному додатку.

2. Більш низька продуктивність: може бути повільним для додатків із високою завантаженістю у випадку слабкого підключення до Інтернету

3. Іноді збільшується витрата акумулятора для складних додатків.

Нативний тип розробки все ще знаходиться на вершині, оскільки набір функцій платформи постійно оновлюється, що робить гібридну розробку складнішою. Переваги нативних по відношенню до гібридних мобільних додатків - це краща продуктивність і UX. Нативна розробка краще підходить для високо інтерактивних додатків з великою кількістю графіки та анімації або коли вміст потрібно швидко оновлювати. Тут ми маємо на увазі ігри, деякі типи соціальних мереж тощо. У випадку, якщо потрібні деякі ексклюзивні можливості, гібридна розробка може вимагати додаткових зусиль та консультації розробників мови платформи, щоб написати їх з нуля.

Гібриди з'явилися з метою пришвидшення розробки. Використання крос-платформ обмежується лише посібниками для мобільних платформ, які накладають свої конструкції та оновлення ОС. Велика перевага полягає в тому, що цей вид розробки дешевший. Гарно розроблений гібридний додаток не має видимих чи функціональних відмінностей для користувачів. Загалом же кінцевому користувачу не важливо як побудований додаток якщо він зручний та приємний у використанні.

Для розробки під ОС Android є три основні IDE: Eclipse (на сьогоднішній день використовується все рідше), IntelliJ idea, Android Studio. В якості інструмента збірки проекту є Ant. Дане IDE легко налаштовується, інтегрується з Heax для розробки з компонентами, такими як SDK Android, NDK та Java машиною. Після успішної інтеграції створити свій проект легко. В якості мови програмування використовується Java. Дане IDE дуже просте в роботі, тому воно підходить для початкового рівня розробки. IntelliJ IDEA - більш серйозний інструмент розробки. Надає такі корисні речі як закриття дужок після умови, груповий перезапис методів та автоматичне створення шаблонних класів (Interface, Singleton). Також тут є можливість зміни теми оформлення. Однак, варто відмітити, що даний проект не є відкритим, однак оновлення виходять дуже часто. Перша версія IntelliJ IDEA з'явилася у січні 2001 року й швидко здобула популярність, як перша Java IDE із широким набором інтегрованих інструментів для рефакторингу, що дозволяла програмістам швидко реорганізовувати сирцевий код програм. Дизайн середовища орієнтовано на продуктивність праці програмістів, дозволяючи їм сконцентруватися на розробці функціональності, тоді як IntelliJ IDEA бере на себе виконання рутинних операцій.

Починаючи з шостої версії продукту IntelliJ IDEA надає інтегрований інструментарій для розробки графічного користувацького інтерфейсу.

З версії 9.0 є безкоштовний варіант Community Edition з відкритими кодами. Сирцеві коди відкритої версії IntelliJ IDEA Community Edition поширюються рамках ліцензії Apache 2.0. Бінарні пакунки підготовлені для Linux, Mac OS X і Windows.

До складу IntelliJ IDEA включені напрацювання, створені в результаті спільної роботи з компанією Google, яка використовувала IntelliJ IDEA як базис для своєї нового відкритого середовища розробки Android Studio. Завдяки співпраці істотно розширені штатні можливості IntelliJ IDEA з розробки застосунків для платформи Android.

Android Studio прийшло на зміну плаґіну ADT для платформи Eclipse. Середовище побудоване на базі вихідного коду продукту IntelliJ IDEA Community Edition, що розвивається компанією JetBrains. Android Studio розвивається в рамках відкритої моделі розробки та поширюється під ліцензією Apache 2.0.

Середовище надає засоби для розробки застосунків не тільки для смартфонів і планшетів, але і для носимих пристроїв на базі Android Wear, телевізорів (Android TV), окулярів Google Glass і автомобільних інформаційно-розважальних систем (Android Auto). Для застосунків, спочатку розроблених з використанням Eclipse і ADT Plugin, підготовлений інструмент для автоматичного імпорту існуючого проекту в Android Studio.

Середовище розробки адаптоване для виконання типових завдань, що вирішуються в процесі розробки застосунків для платформи Android. У тому числі у середовище включені засоби для спрощення тестування програм на сумісність з різними версіями платформи та інструменти для проектування застосунків, що працюють на пристроях з екранами різної роздільності (планшети, смартфони, ноутбуки, годинники, окуляри тощо). Крім можливостей, присутніх в IntelliJ IDEA, в Android Studio реалізовано кілька додаткових функцій, таких як нова уніфікована підсистема складання, тестування і розгортання застосунків, заснована на складальному інструментарії Gradle і підтримуюча використання засобів безперервної інтеграції.

Для прискорення розробки застосунків представлена колекція типових елементів інтерфейсу і візуальний редактор для їхнього компонування, що надає зручний попередній перегляд різних станів інтерфейсу застосунку (наприклад, можна подивитися як інтерфейс буде виглядати для різних версій Android і для різних розмірів екрану). Для створення нестандартних інтерфейсів присутній майстер створення власних елементів оформлення, що підтримує використання шаблонів. У середовище вбудовані функції завантаження типових прикладів коду з GitHub.

До складу також включені пристосовані під особливості платформи Android розширені інструменти рефакторингу, перевірки сумісності з минулими випусками, виявлення проблем з продуктивністю, моніторингу споживання пам'яті та оцінки зручності використання. У редактор доданий режим швидкого внесення правок. Система підсвічування, статичного аналізу та виявлення помилок розширена підтримкою Android API.

Базовою IDE для iOS є Xcode. На відміну від відкритої ОС Android, Ios накладає своєрідну «монополію» на вибір засобів для розробника. Тим не менш, дана IDE по праву займає переше місце серед інструментів для розробки додатків. Це один із небагатьох інструментів, з яким приємно працювати. Xcode використовує єдине вікно робочого простору – workplace window, що містить більшу чакстину необхідних для роботи даних. З Xcode можна легко перейти від написання коду до його відлагодження та розробки користувацького інтерфейсу в межах одного робочого вікна. SDK Ios постійно розширює комплект інструментів Xcode, добавляючи компілятори та фреймворки, необхідні для роботи з ОС. В якості мови для розробки виступає Objective-C – об'єктно-орієнтована мова для розробки всіх Ios-додатків.

ВИСНОВКИ

Виходячи з проведеного аналізу, можна стверджувати, що нативні додатки, які дають найкращу продуктивність, добре підходять для комплексних проектів, що працюють зі складною графікою (наприклад, відеоігри). Неправильний вибір засобу розробки у такому випадку точно дасть про себе знати у кінцевому продукті. У кращому випадку можна отримати невдоволеність користувачів, а у гіршому – взагалі не буде змоги впровадити такий застосунок.

Що стосується менш вимогливих додатків, при їх розробці на кілька платформ (наприклад, iOS та Android) доцільно звернути увагу на такі засоби як Xamarin чи React Native. Їх кінцевий продукт, що мати-

ме нативний користувацький інтерфейс, буде поводитись швидко та плавно при правильному застосуванні технологій. А користувач не помітить жодної втрати продуктивності без спеціальних тестів. Крім того, це суттєво скорочує термін розробки, а відповідно і вартість.

ЛІТЕРАТУРА

1. Nader Dabit. React Native in Action. 2016. P. 64-70.
2. Stan Bershadskiy, Crysfel Villa. React Native Cookbook. 2018. P. 89-92.
3. Jonathan Peppers. Xamarin Cross-platform Application Development: Second Edition. 2015. P. 104.
4. Mahesh Panhale. Beginning Hybrid Mobile Application Development. 2015. P. 110.
5. Rahat Khanna, Sani Yusuf, Hoc Phan. Ionic: Hybrid Mobile App Development. 2017. P. 64-65.
6. Mark L. Murphy. The Busy Coder's Guide to Android Development. 2014. P. 157.
7. Christian Keur, Aaron Hillegass. iOS Programming: The Big Nerd Ranch Guide. 2015. P. 32.
8. Native whis hybrid mobile app development: what to choose in 2018 https://greenice.net/native-vs-hybrid-mobile-app-development-choose/ (viewed on January 17, 2018).
9. React vs. Cordova, PhoneGap, Ionic etc.: https://learnreact.design/2018/02/14/react-native-vs-cordova-phone-gap-ionic-etc/.
10. React native VS Ionic:2 Comparsion: https://medium.com/swlh/react-native-vs-ionic-2-comparison-50aba900be6c (viewed on April 13, 2017). – Title from the screen.

Рецензент: д.т.н., проф. Р.В. Сорокатий
Хмельницький національний університет