УДК 378.14:004.9
DOI https://doi.org/10.32782/2663-5941/2023.4/04

Василенко В.М.
Національний авіаційний університет

Карпенко М.І.
Національний університет харчових технологій

Пуцик М.С.
Національний авіаційний університет

Гуйда О.Г.
Таврійський національний університет імені В.І. Вернадського

РОЗРОБКА ДИЗАЙНУ МОБІЛЬНОГО ДОДАТКУ ДЛЯ ПОКРАЩЕННЯ ЯКОСТІ ОСВІТНЬОГО ПРОЦЕСУ

Ця стаття присвячена розробці дизайну мобільного додатку з метою покращення якості освітнього процесу. У сучасному світі мобільні технології стають все більш актуальними у сфері освіти. Стаття розглядає ключові аспекти розробки дизайну додатку, який допомагатиме вчителям та студентам ефективніше взаємодіяти під час навчання. У статті вивчаються сучасні підходи до розробки інтерфейсу мобільних додатків, враховуючи зручність використання, навігацію та візуальну привабливість. Автори зосереджуються на важливості адаптації дизайну до потреб користувачів з різними рівнями навчальної підготовки та віковими групами. Також розглядаються можливості використання інтерактивних елементів, графіки та мультимедійних ресурсів для підвищення зацікавленості та залученості студентів до навчального процесу. Особлива увага приділяється врахуванню принципів UX-дизайну для створення зручного та задовільного досвіду користувача. Застосування розробленого мобільного додатку може сприяти покращенню комунікації між вчителями та студентами, забезпечити доступ до актуальної інформації, завдань та ресурсів у будь-який час із будь-якого місця. Ця стаття є корисним джерелом для фахівців у галузі освіти та розробки програмного забезпечення, які зацікавлені в покращенні освітнього процесу через використання мобільних технологій. Розробка програмного забезпечення для мобільного додатку вищого навчального закладу є актуальною і значущою задачею у сучасній освітній сфері. Зростаюча популярність мобільних пристроїв серед студентів та педагогічних працівників вимагає створення зручних та ефективних інструментів для поліпшення навчального процесу та комунікації. Створення мобільних додатків передбачає вибір мови програмування, яка найкраще відповідаєзаданим вимогам і цілям, було проаналізовано мови програмування JavaScript, Figma, Native підхід та мова Kotlin.

Ключові слова: *Native development, фреймворки, програмування, JavaScript, Figma, Kotlin, дизайн, мобільний застосунок, освітній процес.*

Постановка проблеми. Широке розповсюдження мобільних пристроїв призвело до розробки багатьох мобільних додатків, спрямованих на покращення якості навчального процесу. Метою розробки застосунку є проектування та дизайн мобільного додатку, який дозволить підвищити якість навчального процесу. В останні роки спостерігається значне зростання використання мобільних додатків в освіті. Мобільні додатки стали потужним інструментом, який може допомогти викладачам та студентам у навчальному процесі. Запропонований мобільний додаток спрямований на підвищення якості навчального процесу шляхом надання студентам зручного та ефективного способу доступу до навчальних процесів та взаємодії з викладачами та одногрупниками. Використання мобільних додатків в освіті привертає значну увагу дослідників в останні роки. Кілька досліджень показали, що використання мобільних додатків в освіті може покращити залучення студентів, посилити співпрацю та полегшити набуття знань і навичок [1]. Окрім того, значна кількість користувачів обирають саме мобільну платформу за рахунок високої мобільності та доступності [2]. Розробка мобільного додатку, здатного вдосконалити навчальний

процес, є особливо актуальною в умовах війни, коли більшість навчальних закладів перейшли на онлайн-навчання.

Аналіз останніх досліджень і публікацій. Для досягнення цієї мети, було проведено детальний огляд періодичних наукових видань, вийшли за останні 5–10 років. Основні результати аналізу попередніх досліджень включають: Tashfeen Ahmad з університету Вест-Індії «Сприйняття студентами використання мобільних телефонів як засобів навчання» [1] присвячена дослідженню впливу інтеграції мобільних телефонів в навчальну діяльність. В роботі колективу науковців з університету Бебкок, яку опублікували в Американському науковий журналі для техніки, технології науки (ASRJETS) [3] проведений детальний порівняльний аналіз стратегій розробки, у якому описуються всі складнощі native, web та hybrid архітектур. У статті Robin Nunkesser з University of Applied Sciences в Хаммі [4] представлено нову таксономію (таксономія – це ієрархічно збудована система цілей і результатів від простої до складної системи) архітектур: на додачу до трьох основних native, web та hybrid запропоновано Endemic Apps, Web Apps, Hybrid Web Apps, Hybrid Bridged Apps, System Language Apps та Foreign Language Apps. Наведено про переваги та недоліки кожного підходу. Праця Ben Frain четвертого видання адаптивного веб-дизайну із HTML5 і CSS [5] докладно описує Responsive Web Design (RWD), а саме Front-End Web Development (увесь синтаксис, роботу з CSS, HTML, медіа, контейнерами та іншими важливими моментами Front-End розробки). Стаття з 13 Іберійської конференції з інформаційних систем і технологій (CISTI) [6] представляє нову концепцію прогресивного веб-додатку, створену Google, щоб нормалізувати всі веб-розробки; описано поточний стан веб-технологій; представлено основні переваги веб-додатків у порівнянні з native підходом.

Виклад основного матеріалу. Для досягнення максимальних результатів та мінімізації витрат необхідно керуватися основами системного аналізу. По-перше було проведено опитування студентів та викладачів Національного авіаційного університету (НАУ), щоб визначити їхні потреби та очікування щодо мобільного додатку для освіти. В результаті опитування було розроблено прототип мобільного додатку, який був протестований групою студентів та викладачів кафедри комп'ютерних технологій дизайну і графіки НАУ. Відгуки цієї групи були використані для доопрацювання додатку та покращення його функцій. Запропонований мобільний додаток був розроблений з використанням комбінації різних методологій. По-друге, було проведено огляд існуючих мобільних додатків, спрямованих на підвищення якості навчального процесу, таких як Google Classroom, Microsoft teams, Quizlet тощо. Цей огляд дозволив виявити сильні та слабкі сторони існуючих додатків та визначити найбільш важливі функції, які повинні бути включені в запропонований додаток, а саме:

Головна сторінка: можливість побачити за яким тижнем зараз відбувається навчання; швидкий доступ до домашніх завдань; можливість швидко переглянути, які пари будуть завтра; таймер, що вказує на те, через скільки часу розпочнеться наступна пара; кнопка «Перейти до профілю»:

- Розклад: розклад занять за вибором тижня та дня тижня; лекції виділяються синім кольором, а практичні завдання – ні; розклад дзвінків.
- Домашні завдання: можливість внесення домашніх завдань у список; можливість редагувати домашні завдання.
- Контакти: представлено прізвище викладача, його ініціали, дисципліну, яку він викладає, номер телефону та аватар; представлено прізвище одногрупника, його ініціали, номер телефону та аватар (свій номер студенти та викладачі вказують при реєстрації, а аватари автоматично синхронізуються з корпоративної пошти університету);
- Профіль: представлена особиста інформація – так звана «картка студента»: його корпоративна пошта, ПІБ, номер телефону, назву факультету, на якому він навчається, назва його групи, номер його підгрупи, та ім'я його куратора; кнопка «налаштування»; кнопка «редагувати профіль», де можна змінити свій аватар, номер телефону тощо; можливість увімкнути темну тему.

По-третє, було розроблено прототип мобільного додатку, який був протестований групою студентів та викладачів. Відгуки цієї групи були використані для доопрацювання додатку та покращення його функцій.

Дизайн був розроблений у графічному редакторі Figma. Це зручний графічний редактор, в якому можна створювати: прототипи web-сайтів і додатків; окремі елементи інтерфейсу: іконки, кнопки, форми і багато іншого; векторні зображення та ілюстрації, інше. Основною зручністю є багатокористувацький режим редагування: досить відправити посилання на проект, відкривши доступ для редагування, і інший дизайнер зможе вносити в нього свої зміни.

Інтерфейс розробленого мобільного додатку для вищого навчального закладу повинен бути

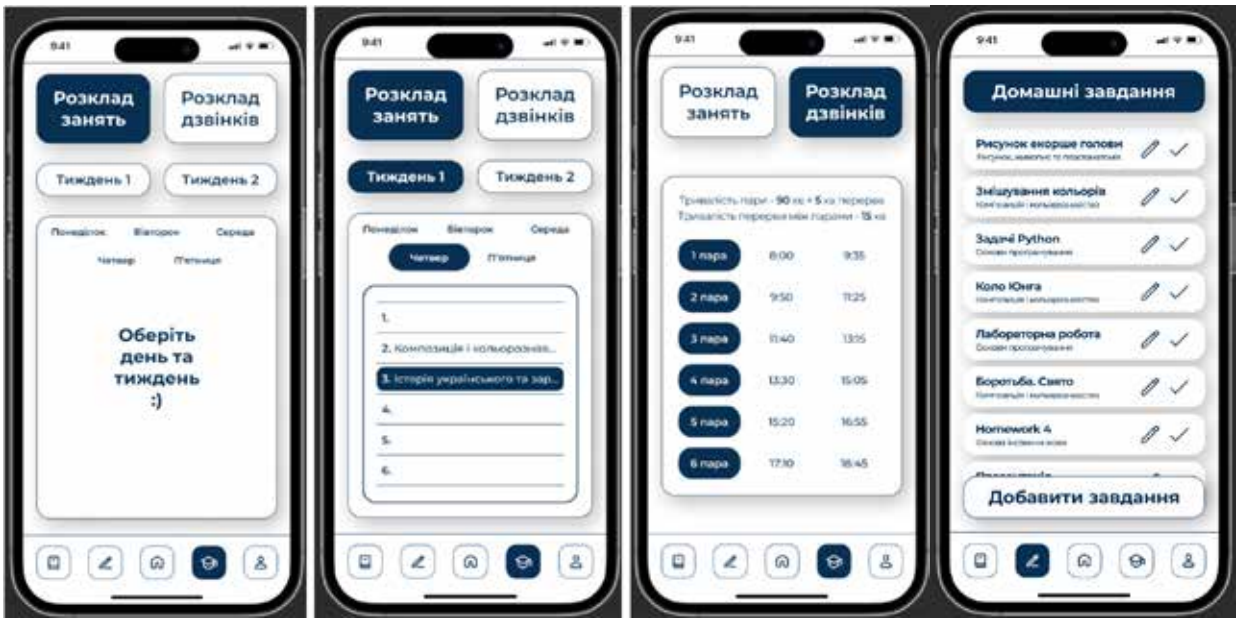

Рис. 1. Прототип інтерфейсу мобільного додатку для студентів НАУ

добре продуманим, зручним та легким у використанні. Ось деякі ключові аспекти враховувати при проектуванні інтерфейсу:

1. **Простота та Інтуїтивність:** Інтерфейс повинен бути легким у використанні навіть для тих, хто не має досвіду з подібними додатками. Головні функції мають бути доступні на перший погляд.

2. **Зручна навігація:** Забезпечте логічну інформаційну структуру, яка дозволить користувачам швидко знаходити потрібні розділи та ресурси. Використовуйте меню, вкладки або інші елементи навігації.

3. **Адаптація до різних пристроїв:** Розгляньте різні розміри екранів та орієнтації пристроїв (горизонтальна/вертикальна). Інтерфейс повинен добре виглядати та працювати на різних пристроях.

4. **Привабливий дизайн:** Використовуйте приємні кольори, адекватний шрифт та чітку графіку, щоб зробити інтерфейс привабливим і сприяти комфортному читанню та взаємодії.

5. **Інтерактивність:** Додайте інтерактивні елементи, такі як кнопки, покажчики, анімацію тощо. Вони можуть поліпшити взаємодію та зробити додаток цікавішим.

6. **Персоналізація:** Дайте користувачам можливість налаштувати деякі аспекти додатку, наприклад, обрати певний стиль відображення інформації або налаштувати сповіщення.

7. **Оптимізація швидкості:** Забезпечте швидку реакцію додатку на дії користувача, щоб уникнути незручностей і часових затримок.

8. **Спеціальні функції:** Розгляньте можливість додати додаткові функції, такі як розклад занять, електронний журнал, можливість завантаження матеріалів тощо.

9. **Тестування:** Проведіть тестування інтерфейсу з реальними користувачами, щоб виявити можливі недоліки та покращити його перед випуском.

Актуальність розробки програмного забезпечення для мобільних додатків для вищих навчальних закладів визначається декількома факторами:

1. **Зручність та доступність:** Студенти та викладачі все частіше використовують мобільні пристрої як головний спосіб отримання інформації. Мобільний додаток надає зручний доступ до розкладу занять, матеріалів для навчання, завдань та інших ресурсів в будь-який час і з будь-якого місця.

2. **Покращення комунікації:** Мобільний додаток може стати платформою для взаємодії між викладачами та студентами. Це сприяє швидкій обмін інформацією, спільному вирішенню питань та підтримці.

3. **Оптимізація навчання:** Використання мультимедійних ресурсів, інтерактивних елементів та графіки може поліпшити сприйняття навчального матеріалу та зробити навчання більш ефективним.

Проте, розробка програмного забезпечення для мобільного додатку вищого навчального закладу також вносить свої виклики:

1. **Різноманітність платформ:** Розробка додатку для різних мобільних платформ (iOS, Android тощо) може потребувати додаткового зусилля та ресурсів.

2. **Безпека та конфіденційність:** Зберігання та обробка особистої інформації студентів і викладачів вимагає високого рівня кібербезпеки.

3. **Унікальні потреби:** Різні вищі навчальні заклади мають свої особливі вимоги та потреби, що може ускладнити процес розробки універсального додатку.

4. **Технічна складність:** Врахування різноманітних функцій, інтерфейсів та можливостей може створити технічні труднощі.

Усі ці аспекти потребують ретельного планування, проектування та розробки, але успішно реалізований мобільний додаток може відкрити нові можливості для покращення якості освітнього процесу, комунікації та залучення студентів до навчання

Загалом, інтерфейс мобільного додатку повинен бути зорієнтований на забезпечення зручного та приємного користування, а також відповідати особливостям навчальної діяльності та потребам користувачів вищого навчального закладу. Наступним кроком був вибір напрямку розробки. На момент сьогодення існує три шляхи розробки мобільних додатків:

Native – передбачає використання офіційних мов програмування та інструментів, які надаються платформою (наприклад, Java або Kotlin для Android та Objective-C або Swift для iOS). Native надає повний функціонал пристрою, який обмежений лише вміннями розробника та його уявою; забезпечує найвищу продуктивність та бажаний вигляд створюваного додатку [10]. Переваги native розробки: висока продуктивність; повний функціонал та інтерфейс; Недоліки native розробки: витрати на розробку та обмеження платформи (даний підхід використовується лише для однієї платформи. Якщо потрібен застосунок для іншої, то все потрібно починати з нуля); складність підтримки.

Hybrid – передбачає використання двигуна браузера в мобільному пристрої, вбудовуючи HTML-контент у native веб-контейнер (наприклад, WebView для Android, UIWebView для iOS). Гібридні додатки запускаються внутрішніми браузерами, які доступні на платформі. Фреймворки, такі як Apache Cordova (також відомий як PhoneGap), React Native, Ionic, дозволяють розробляти гібридні додатки, які можуть працювати як на Android, так і на iOS. Переваги гібридної розробки: можна поширювати через Appstore; поширюється на будь-яку платформу; гібридні програми можна застосовувати як для серверних, так і для автономних програм; гібридні програми можуть отримати доступ до native функцій (наприклад, камера, мікрофон і т.д.). Недоліки гібридної розробки: слабша продуктивність (за рахунок використання двигуна браузера), на додачу міст JavaScript накладає додаткові витрати на продуктивність під час доступу до API конкретної платформи пристрою; однакова взаємодія на всіх платформах (існує певна неузгодженість, якщо наприклад, деякі пристрої мають фізичну кнопку «Назад», а на деяких телефонах кнопкою «Назад» можна керувати на екрані); велика залежність від інтернету; транслювання обмежене мостом JavaScript [3].

Web – передбачає використання стандартних веб-технологій, таких як HTML, CSS та JavaScript. Розроблені таким чином додатки по суті є мобільними версіями веб-сайтів у браузері пристрою та не вимагають упакування у контейнер додатків. Переваги web розробки: однакова взаємодія зі всіма платформами; відсутня необхідність оновлень; легке вивчення та розробка (висока швидкість розробки) за допомогою web технологій; один застосунок для всіх платформ; використання обробки лише на сервері, а не на пристрої користувача; більш портативна за native. Недоліки web розробки: обмежений доступ до функцій та функціоналу (немає можливості підключити контакти або камеру та мікрофон і т.п.); залежність від інтернету; доступ лише через URL-адреси (а не через магазин програм); менша продуктивність у порівнянні з native архітектурою [3,4].

Окрім того існують фреймворки (frameworks – набір готових інструментів, правил і шаблонів, які допомагають програмістам розробляти програми або веб-додатки) та маніфести (manifest – обов'язковий файл у форматі XML, який містить інформацію про додаток, використовується для представлення важливих метаданих про додаток), які можна (і потрібно) використовувати в процесі розробки. Використання фреймворків також має як переваги, серед яких варто виділити: пришвидшення розробки, кросплатформність (можливість застосування додатку більше, ніж на одній операційній системі), спільний код (один код можна використовувати для різних платформ), велику спільноту; так і недоліки: обмеження функціональності (в тому числі обмеження на доступ до низькорівневих системних функцій), залежність додатку від фреймворку, додаткова вага додатку та невисока швидкодія, обмеження можливості UI/UX (User Interface – зовнішній вигляд та інтерактивний дизайн додатку або веб-сайту; User Experience – загальні враження користувача від

взаємодії з продуктом) (фреймворки можуть мати обмежені можливості для налаштування вигляду та взаємодії з користувачем. Це може призвести до обмеженого контролю над UI/UX додатком, особливо якщо потрібен унікальний та індивідуальний дизайн).

Web розробка розбивається на дві категорії, а саме Responsive Web Design (RWD) – це підхід до верстки веб-сторінок, який створює оптимальне відображення та взаємодію веб-сторінок на різних пристроях і розмірах екранів. Основна мета RWD полягає в тому, щоб веб-сторінки були легко читабельними, навігація була зручною і елементи контенту пристосовувалися до розміру екрану, забезпечуючи оптимальний користувацький досвід незалежно від пристрою [5].

Progressive Web Apps (PWA): – це підхід до розробки мобільних пристроїв, спрямований на подолання труднощів і недоліків попередніх архітектур. Застосування цього підходу створює особливий тип веб-програм, які не потребують встановлення перед використанням і обслуговуються з віддаленого сервера через захищений протокол передачі гіпертексту (HTTPS), на відміну від звичайних мобільних веб-програм, які можуть обслуговуватися за допомогою HTTP [3]. Різниця між PWA та гібридним способом розробки полягає в наступних речах: повна підтримка екрану (без браузера), миттєве завантаження push-сповіщень навіть в офлайн стані.

Переваги PWA: швидке завантаження; мала вага (в середньому 2 МБ); можливість індексації (наприклад, можливість відстежити певні дії користувача, щоб створити адаптивний веб-дизайн); можливість використання push-сповіщень.

Недоліки PWA: відсутність можливостей App Store; високе енергоспоживання; труднощі з доступом до певних функцій (наприклад, PWA не можуть отримати доступ до NFC пристрою, Bluetooth, камери і т.п.).

Варто уточнити що мається на увазі під web-програмою. Різниця між веб-сайтом та веб-програмною досить розмита. Веб-програма – це взагалі більше веб-сайт, ніж програма, однак сутність сайту більше полягає у відображенні, а програми у взаємодії з користувачем. Наприклад, сайт новин буде «веб-сайтом», а електронна таблиця або спільний календар будуть веб-«додатками». Найпоширенішим прикладом веб-програм є програми Google, такі як Gmail, Google Docs, Spreadsheets і Slides. Підсумовуючи перелічені вище факти необхідно провести аналіз для визначення найкращого підходу відповідно до нашого конкретного випадку. Нашою основною метою є розробка мобільного додатку для студентів та викладачів з розкладом, контактами та домашніми завданнями.

Оскільки перед нами стоять вимоги перед дизайном, розглядаємо наступні можливі шляхи реалізації: нативний (native), гібридний (hybrid), веб-додаток (web), прогресивний веб-додаток (PWA) та адаптивний дизайн (RWD). Нативний підхід дозволяє реалізувати будь-який дизайн додатку, але вимагає складної розробки для кожної платформи. Інші архітектури обмежені так чи інакше.

Гібридна розробка обмежена фреймворками (гібридні рішення апріорі програють нативним рішенням), неможливо зробити більш-менш складний UI: наприклад, браузер може вирішити, що необхідно перемалювати сторінку і весь інтерфейс починає блимати при навігації, а також гібридні фреймворки виходять з затримкою відносно нових версій операційних систем і тоді постає питання чекати оновлень або підтримувати застосунок власноруч. Дзеркальна ситуація до повільного випуску оновлень: якщо не ставити фіксовані версії плагінів, внаслідок оновлень білд мобільного застосунку може просто «розсипатись» (перестати запускатись).

PWA – це веб-сайт, створений за допомогою веб-технологій, який діє як додаток і не потребує встановлення як нативний додаток. Він відносно легко реалізовується, однак така архітектура має наступні недоліки: обмежені можливості фонової роботи (немає повного контролю над фоновою роботою і поведінкою після закриття додатку); не має прямого доступу до деяких функцій пристроїв, таких як контакти, календар, Bluetooth, NFC та інші (деякі з цих функцій можуть бути доступні через веб-API, однак їх використання може бути обмеженим); дизайн може відрізнятися на різних платформах; PWA потребує доступу до Інтернету для своєї роботи. Вони можуть працювати в офлайн-режимі, але обмежені функціональні можливості можуть бути доступні без Інтернет-з'єднання [7]; реалізувати PWA можна на більшості, але не на всіх пристроях [8].

Web – додатки та RWD побудовані на базі веб-технологій, таких як HTML, CSS та JavaScript, і мають обмежений доступ до функцій пристрою; обмеженість офлайн режиму, залежність від браузера та інтернету. Звичайно, завжди можна зробити щось інакше, додати елементи, які будуть реалізовувати непідйомні для архітектур задачі, такі елементи називаються «костилі». Проте чим

складніше проект, тим більше «костилів» знадобиться [9, 10]. Таким чином, ми прийшли до висновку, що native підхід є найкращим з огляду на вимоги щодо дизайну.

Розробка Native додатку пропонується реалізувати на статично типізованій мові Kotlin. Однією з найбільш істотних відмінностей між Kotlin і Java є синтаксис. Kotlin має більш стислий синтаксис, ніж Java, що означає, що для виконання тих самих операцій потрібно менше коду. І Kotlin, і Java компілюються в байт-код, який працює на JVM, що означає, що вони мають подібні характеристики продуктивності. Однак, в деяких випадках Kotlin має деякі переваги в продуктивності перед Java. Наприклад, функція нульової безпеки Kotlin може допомогти зменшити кількість винятків під час виконання та покращити загальну продуктивність програми. Kotlin підтримує лямбда-вирази, функції вищого порядку та функції розширення. Сукупність цих властивостей робить дану мову програмування ідеальним варіантом для створення мобільного додатку.

Висновки. Створено унікальний дизайн мобільного додатку, були виявлені сильні та слабкі сторони існуючих стратегій розробки, а також визначені та застосовані найбільш важливі функції, які повинні бути включені в додаток. В результаті аналізу визначено найбільш ефективні способи та засоби розробки мобільних додатків: Native підхід та мова Kotlin. Запропонований мобільний додаток має потенціал для підвищення якості освітнього процесу шляхом надання студентам зручного та ефективного способу доступу до навчальних матеріалів та взаємодії з викладачами та одногрупниками. Додаток простий у використанні та доступний з будь-якого місця і в будь-який час. Додаток можна вдосконалювати, додаючи додаткові функції.

Список літератури:

1. Ahmad T. Student perceptions on using cell phones as learning tools. PSU Research Review. 2020. Vol. 4, no. 1. P. 25–43. URL: https://doi.org/10.1108/prr-03-2018-0007 (дата звернення: 21.08.2023).

2. Pal D., Vanijja V. Perceived usability evaluation of Microsoft Teams as an online learning platform during COVID-19 using system usability scale and technology acceptance model in India. Children and Youth Services Review. 2020. Vol. 119. P. 105535. URL: https://doi.org/10.1016/j.childyouth.2020.105535 (дата звернення: 21.08.2023).

3. Adetunji, O., Ajaegbu, C., Otuneme, N., & Omotosho, O. J. Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development. Am. Sci. Res. J. Eng. Technol. Sci.(ASRJETS). 2020. Vol. 68(1). P. 85–99.

4. Nunkesser R. Beyond web/native/hybrid. ICSE '18: 40th International Conference on Software Engineering, Gothenburg Sweden. New York, NY, USA, 2018. URL: https://doi.org/10.1145/3197231.3197260 (дата звернення: 21.08.2023).

5. Frain, B. Responsive Web Design with HTML5 and CSS: Build future-proof responsive websites using the latest HTML5 and CSS techniques. Packt Publishing Ltd. 2022. 498 p.

6. Fortunato D., Bernardino J. Progressive web apps: An alternative to the native mobile Apps. 2018 13th Iberian Conference on Information Systems and Technologies (CISTI), Caceres, 13–16 June 2018. 2018. URL: https://doi.org/10.23919/cisti.2018.8399228 (дата звернення: 21.08.2023).

7. Tandel S.S., Jamadar A. Impact of Progressive Web Apps on Web App Development. International Journal of Innovative Research in Science, Engineering and Technology (A High Impact Factor, Monthly, Peer Reviewed Journal). 2018. Vol. 7. Issue 9. P. 9439–9444.

8. Biørn-Hansen A., Majchrzak T. A., Grønli T.-M. Progressive Web Apps: The Possible Web-native Unifier for Mobile Development. 13th International Conference on Web Information Systems and Technologies, Porto, Portugal, 25–27 April 2017. 2017. URL: https://doi.org/10.5220/0006353703440351 (дата звернення: 21.08.2023).

9. Shevtsiv N. A., Striuk A. M. Cross platform development vs native development. CS&SE@SW 2020: 3rd Workshop for Young Scientists in Computer Science & Software Engineering, November 27 2020. Kryvyi Rih. 2020. Vol. 2832. P. 75–83.

10. Android Application Development: A Brief Overview of Android Platforms and Evolution of Security Systems / A. Sarkar et al. 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 12–14 December 2019. 2019. URL: https://doi.org/10.1109/i-smac47947.2019.9032440 (дата звернення: 21.08.2023).

Vasylenko V.M., Karpenko M.I., Putsyk M.S., Guida O.G. DEVELOPMENT OF A MOBILE APPLICATION DESIGN TO IMPROVE THE QUALITY OF THE EDUCATIONAL PROCESS

This article is devoted to the development of the design of a mobile application in order to improve the quality of the educational process. In today's world, mobile technologies are becoming increasingly relevant in the field of education. The article examines the key aspects of developing an application design that will help teachers and students interact more effectively during learning. The article examines modern approaches to the development of mobile application interfaces, taking into account usability, navigation and visual appeal. The authors focus on the importance of adapting the design to the needs of users with different educational levels and age groups. The possibilities of using interactive elements, graphics and multimedia resources to increase the interest and involvement of students in the educational process are also considered. Special attention is paid to taking into account the principles of UX design to create a convenient and satisfactory user experience. The use of the developed mobile application can help improve communication between teachers and students, provide access to relevant information, tasks and resources at any time from any place. This article is a useful resource for education and software development professionals interested in improving the educational process through the use of mobile technologies. The development of software for a mobile application of a higher educational institution is a relevant and significant task in the modern educational sphere. The growing popularity of mobile devices among students and teaching staff requires the creation of convenient and effective tools to improve the educational process and communication. The creation of mobile applications involves the selection of a programming language that best meets the given requirements and goals, the programming languages JavaScript, Figma, Native approach and the Kotlin language were analyzed.

Key words: *native development, frameworks, programming, JavaScript, Figma, Kotlin, design, mobile application, educational process.*