Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ГОЛОВНОЙ УЧЕБНО-ИССЛЕДОВАТЕЛЬСКИЙ И МЕТОДИЧЕСКИЙ ЦЕНТР ПРОФЕССИОНАЛЬНОЙ РЕАБИЛИТАЦИИ ЛИЦ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ (ИНВАЛИДОВ)

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

Исследование средств для разработки мобильных приложений

Студент группы ИУ5Ц-92Б _____ М.В. Яровенко
(код группы) (подпись, дата) (инициалы и фамилия)

Научный руководитель _____ К.С. Мышенков
(подпись, дата) (инициалы и фамилия)

Оценка _____

Руководитель от кафедры _____ В.И. Терехов
(подпись, дата) (инициалы и фамилия)

Оценка _____

Москва, 2023 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой ИУ5

_____ В.И. Терехов

« ____ » _____ 2023 г.

ЗАДАНИЕ
на выполнение научно-исследовательской работы

по теме: Исследование средств для разработки мобильных приложений

Студент группы ИУ5Ц-92Б Яровенко Максим Васильевич
(Фамилия имя отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.) исследовательская

Источник тематики (кафедра, предприятие, НИР) учебная тематика

График выполнения НИР: 25% к 5 нед., 50% к 9 нед., 75% к 13 нед., 100% к 16 нед.

***Техническое задание*:** Исследование существующих средств разработки мобильных приложений; анализ существующих средств разработки мобильных приложений; классификация фреймворков для разработки мобильных приложений.

Оформление научно-исследовательской работы:
Расчетно-пояснительная записка, минимальный объем 12 листов формата А4.
Приложения: графический (иллюстративный) материал (чертежи, схемы, диаграмма и т.п.)

Дата выдачи задания «15» октября 2023 г.

Научный руководитель _____ К.С. Мышенков
(подпись, дата) (инициалы и фамилия)

Студент группы ИУ5Ц-92Б _____ М.В. Яровенко
(код группы) (подпись, дата) (инициалы и фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Содержание

ВВЕДЕНИЕ

С постоянным ростом популярности мобильных устройств и возрастающим спросом на инновационные мобильные приложения, исследование средств для их разработки становится все более актуальным. В разработке успешного мобильного приложения важное значение имеют инструменты, используемые разработчиками. Путем исследования различных средств разработки мобильных приложений можно найти наиболее эффективные и эргономичные инструменты, способствующие разработке высококачественных и интуитивно понятных приложений. В данной работе будут рассмотрены различные средства для разработки мобильных приложений, их особенности, преимущества и недостатки, а также подробно проанализированы примеры использования каждого инструмента. Исследование этих средств позволит разработчикам мобильных приложений выбрать наиболее оптимальное решение для своих проектов и повысить их конкурентоспособность на насыщенном рынке мобильных приложений.

В данном исследовании целью является сравнение и классификация популярных средств разработки мобильных приложений, которые используются разработчиками. Будут сравниваться нативные и ненативные (кросс-платформенные) решения по различным техническим характеристикам, таким как используемый язык программирования, подход к созданию компонентов и целевые операционные системы. Также рассматриваем бизнес-требования, включая легкость поиска разработчика для конкретной технологии, год выпуска инструмента и открытость исходного кода. Наша цель - предоставить рекомендации по выбору инструмента разработки, учитывая проектные требования, бюджет и доступное время для разработки мобильного приложения.

АНАЛИЗ СРЕДСТВ РАЗРАБОТКИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

Рынок разработки мобильных приложений

На данный момент на рынке разработки мобильных приложений отсутствует единый стандартный инструмент (язык или фреймворк) для создания приложений. Некоторые компании выбирают экономию времени и ресурсов, используя кросс-платформенные решения, которых сейчас на рынке представлено больше десятка. Другие компании, чаще всего состоятельные, предпочитают нативную разработку, чтобы иметь полный доступ ко всем функциям операционных систем. На сегодняшний день популярными инструментами для создания клиентских мобильных приложений являются кросс-платформенная разработка с использованием фреймворков React Native, Xamarin, NativeScript, Flutter, Ionic, Cordova, PhoneGap и нативная разработка для iOS и Android [2].

Нативная iOS-разработка

Нативная iOS-разработка - это процесс создания мобильных приложений для устройств, работающих на операционной системе iOS, с использованием официальных инструментов и технологий Apple. В основе нативной разработки лежит Objective-C или Swift - языки программирования, специально разработанные для создания приложений для iOS [6].

Для нативной iOS-разработки применяются различные инструменты, такие как Xcode - интегрированная среда разработки (IDE) для iOS, которая позволяет создавать приложения, отлаживать код, отслеживать версии и выполнять другие задачи разработчика. В Xcode также предоставляется доступ к набору фреймворков, которые разработчик может использовать для добавления различных функций и возможностей в свое приложение [7].

Одним из ключевых преимуществ нативной iOS-разработки является высокая производительность и оптимальное использование аппаратных ресурсов устройства.

Однако нативная разработка требует от разработчика знания специфических инструментов и языков программирования Apple, а также может ограничивать кросс-платформенные возможности приложения.

Разработка и отладка приложений возможна только на компьютерах с операционной системой MacOS, которую можно приобрести, купив MacBook. Чтобы загрузить приложение в магазин AppStore (единственный легальный способ установки приложений в iOS), разработчику необходим Apple Developer Account [3]. Как можно заметить, для начинающих разработчиков финансовый порог в iOS-разработке довольно высок. Однако крупные компании могут позволить себе покрыть эти расходы и сделать приложение с глубокими платформенными интеграциями для получения доступа к самым последним функциям операционной системы iOS и наилучшей производительности на этой платформе.

Нативная Android-разработка

Нативная разработка под Android означает создание приложений, используя официальный набор инструментов и язык программирования, предоставляемые Google. Основной язык программирования для разработки под Android является Java, но также возможно использование Kotlin, который является официальным альтернативным языком.

При разработке нативного приложения под Android, разработчик может использовать Android Software Development Kit (SDK), который предоставляет комплект API (Application Programming Interface), инструментарий и библиотеки для создания, тестирования и отладки приложений под Android [4].

Преимущества нативной разработки под Android включают высокую производительность, полный доступ к функциональности устройства,

возможность работы без подключения к интернету и доступ к огромному сообществу разработчиков.

Однако, разработка нативных приложений под Android требует знания Java или Kotlin, а также дополнительного времени и усилий для создания приложений, соответствующих стандартам Android и рекомендациям Google.

Использование фреймворков

Фреймворки для разработки мобильных приложений на разных платформах были разработаны для замены нативной разработки. Они обладают рядом преимуществ перед нативными инструментами, такими как быстрота и более низкая стоимость разработки. Однако, существуют и недостатки, такие как возможное снижение производительности или ограниченный набор функций для интеграции с операционными системами. Рассмотрим некоторые из таких кросс-платформенных фреймворков, таких как React Native, Xamarin, NativeScript, Flutter, Ionic, Cordova, PhoneGap. Unity не включается в обзор из-за его основного использования в индустрии мобильных игр, а остальные фреймворки не рассматриваются из-за их низкой популярности среди разработчиков. Важно отметить, что для загрузки приложения, созданного с использованием кросс-платформенного фреймворка, в магазины App Store или Google Play, также необходимо иметь аккаунт разработчика на соответствующей платформе.

КЛАССИФИКАЦИЯ ФРЕЙМВОРКОВ

Классификация кросс-платформенных решений

Когда мы говорим о классификации кросс-платформенных решений, мы имеем в виду различия между ними, несмотря на то, что все они позволяют написать код и преобразовать его в установочные файлы для разных операционных систем. Классификацию можно провести на основе того, как работает исполняемый код или как описывается интерфейс. На рисунке 1 показана такая классификация, которая основана на подходе, используемом при разработке кросс-платформенных фреймворков [5].

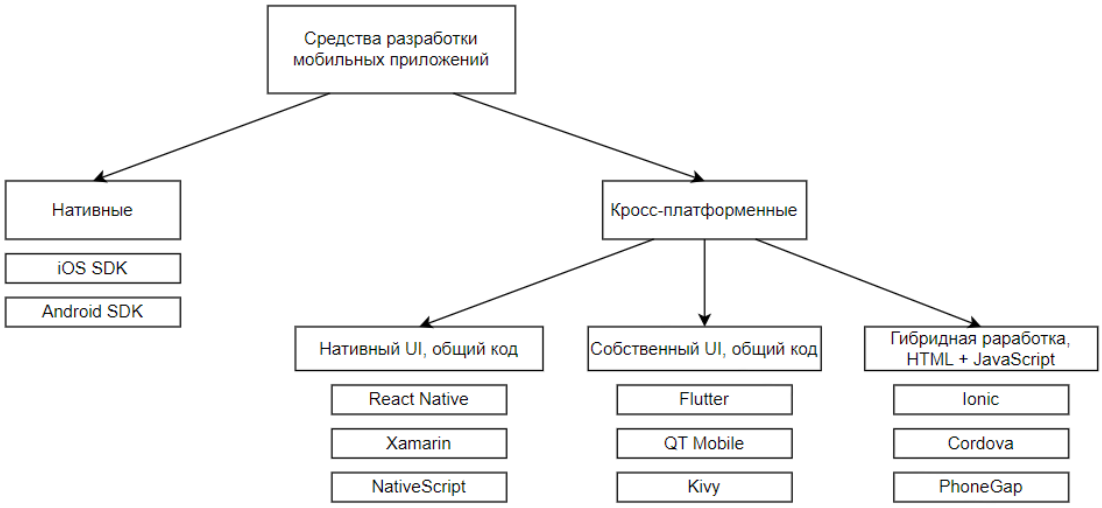

Рисунок 1 – Классификация фреймворков

Нативный User Interface, общий код

Фреймворки такого типа (React Native, Xamarin, NativeScript) подойдут в первую очередь тем, кто хочет добиться, чтобы мобильное приложение выглядело подобно нативному. Эти фреймворки могут быть классифицированы по различным способам.

Классификация по работе исполняемого кода: код может быть компилируемым (как в случае фреймворка Xamarin и используемого в нем языка

C#), так и интерпретируемым с JIT-компиляцией (большинство подобных фреймворков использует JavaScript).

Классификация по способу описания интерфейса: некоторые фреймворки, например Xamarin, используют нативные элементы интерфейса каждой платформы. Другие фреймворки, такие как Xamarin Forms, предлагают универсальные элементы интерфейса (набор виджетов), которые преобразуются в компоненты интерфейса каждой платформы. Также существует способ, при котором реализуется разный интерфейс для разных платформ с сохранением общего подхода (например, React Native создает обёртки вокруг нативных элементов интерфейса. Соответственно, интерфейс описывается для каждой платформы отдельно, но по одному принципу).

Рассмотрим один из популярных фреймворков данной группы - React Native. React Native - это фреймворк, разработанный компанией Facebook в 2015 году, который позволяет создавать мобильные приложения для iOS и Android. Фреймворк использует подход "Нативный UI, общий код", что означает, что разработчики могут использовать нативные компоненты обеих операционных систем, при этом логика мобильного приложения пишется на языке JavaScript. Для взаимодействия с платформой React Native использует так называемый мост. В то время как JavaScript и нативные потоки написаны на совершенно разных языках, эта функция моста делает возможной двунаправленную связь. Это означает, что если уже есть собственное приложение для iOS или Android, то все равно можно использовать его компоненты или перейти к разработке на React Native. React Native является технологией с открытым исходным кодом, что позволяет разработчикам со всего мира вносить свой вклад в ее развитие, и имеет большое сообщество разработчиков. React Native отлично подходит, если вам нужно быстро и недорого создать приложение для iOS и Android, которое будет выглядеть так же, как нативные приложения (будет состоять из нативных компонентов). Однако из-за использования JavaScript могут возникать проблемы с производительностью, особенно при добавлении сложных анимаций.

Собственный User Interface, общий код

Фреймворки данного типа, такие как Flutter, используют общий код для отрисовки пользовательского интерфейса, что позволяет достичь высокой производительности и реализовывать сложные дизайнерские интерфейсы. Однако у таких фреймворков есть некоторые недостатки, такие как больший размер приложений и более высокое потребление оперативной памяти устройства. Также приложения, разработанные с использованием этих фреймворков, могут иметь одинаковый внешний вид на разных операционных системах.

Рассмотрим один из популярных фреймворков данной группы – Flutter. Flutter - это современный фреймворк, разработанный компанией Google в 2017 году. Он позволяет создавать многофункциональные приложения для различных платформ, включая iOS, Android, MacOS, Windows, Linux и веб-приложения. Для написания приложений на Flutter используется язык программирования Dart, который также был разработан в Google и считается более зрелой альтернативой JavaScript. Главной особенностью данного фреймворка является то, что все компоненты пользовательского интерфейса наследуются от одного класса Widget, будь то кнопка, текст или всё приложение, также можно создавать собственные виджеты, что делает возможным построение интерфейсов любой сложности. Код Flutter компилируется в нативный код для каждой платформы, что обеспечивает высокую производительность. Фреймворк использует графический движок Skia. Flutter также имеет доступ к библиотекам и платформенным API, которые доступны в нативных приложениях SDK, с помощью технологии MethodChannel. Исходный код Flutter открыт, и у него уже большое активное сообщество разработчиков, превосходящее по размеру сообщество React Native. Если вам нужно приложение с необычным пользовательским интерфейсом и сложными анимациями, то Flutter будет отличным выбором. Однако следует учесть, что Flutter не поддерживает нативные UI-компоненты из коробки, поэтому если вам

нужно, чтобы ваше приложение выглядело полностью нативно, это может быть недостатком.

Гибридная разработка HTML + JavaScript

Гибридная разработка HTML + JavaScript (Ionic, Cordova, PhoneGap) представляет собой подход, при котором создается веб-страница, открытая во встроенном браузере приложения. Основным преимуществом этого подхода является экономия средств на разработку, так как можно использовать уже имеющийся код и привлечь разработчиков, работавших над веб-приложением. Еще одним плюсом является большое количество доступных библиотек для HTML + JS, что означает отсутствие проблем, связанных с необходимостью создавать собственную версию известной библиотеки. Однако, несмотря на преимущества, есть существенные недостатки, такие как отсутствие поддержки нативных жестов и проблема совместимости с браузерами. На старых версиях Android WebView не обновлялся автоматически, поэтому гибридные приложения не могут использовать новейшие функции браузера на этих устройствах. Это ограничивает минимальную версию Android или требует включения WebView в код приложения, увеличивая его размер. Рекомендуется использовать гибридную разработку только в случае очень ограниченного бюджета или если необходимо повторить функционал существующего веб-сайта, а полнофункциональное мобильное приложение не требуется.

ВЫВОДЫ

Выбор инструмента разработки – важнейший этап в разработке мобильного приложения. В данном исследовании были рассмотрены и проанализированы популярные варианты разработки мобильных приложений. Было проведено сравнение между нативными и ненативными (кросс-платформенными) решениями разработки, учитывая их технические характеристики и соответствие бизнес-требованиям компаний, специализирующихся на создании приложений. Приведенные рекомендации по выбору инструмента разработки могут быть полезны разработчикам, заинтересованным в выборе инструментов разработки, которые помогут быстро создавать современный программный код с учетом структуры, области применения и других факторов, которые обязательно следует учитывать при разработке программного приложения.

Проанализировав доступные средства разработки мобильных приложений, я решил использовать для выполнения дипломной работы фреймворк React Native, так как он прост в освоении и позволяет относительно быстро создать продукт (мобильное приложение), который выглядит подобно нативному.

Список используемых источников

1. Волков А.И. Анализ средств разработки мобильных приложений // Современная техника и технологии. 2017. № 5 [Электронный ресурс]. URL: https://technology.snauka.ru/2017/05/13222 (дата обращения: 20.12.2023).
2. Cross-platform mobile frameworks used by software developers worldwide from 2019 to 2021. Statista. [Электронный ресурс]. URL: https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/ (дата обращения: 20.12.2023).
3. Develop // Apple Developer. [Электронный ресурс]. URL: https://developer.apple.com/ (дата обращения: 20.12.2023).
4. Modern Android Development // Android Developers. [Электронный ресурс]. URL: https://developer.android.com/ (дата обращения: 20.12.2023).
5. Чуриков Е.А., Зудилова Т.В., Ананченко И.В., Осипов Н.А., Иванов С.Е., Осетрова И.С. Сравнение современных средств разработки мобильных приложений // Современные наукоемкие технологии. – 2022. – № 12-1. – С. 82-87; [Электронный ресурс]. URL: https://top-technologies.ru/ru/article/view?id=39441 (дата обращения: 20.12.2023).
6. Марк Д. iOS 5 SDK. Разработка приложений для iPhone, iPad и iPod touch / Д. Марк , Д. Наттинг , Д. Ламарш. - М.: Вильямс, 2012. - 672 с.
7. Swift. Разработка приложений в среде Xcode для iPhone и iPad с использованием iOS SDK. - М.: Вильямс, 2015. - 816 с.
8. Farook F., Galloway M. iOS 11 & Swift For Beginners // Razeware LLC, 2017. – 706 с.
9. Методические рекомендации по подготовке и защите выпускной квалификационной работы бакалавра. / Кротов Ю.Н. [Электронный ресурс] – URL: https://drive.google.com/file/d/1pEcfTr3xDdJ81Hxz2F6GcbtNV1n3dan6/view. (дата обращения: 20.12.2023).