Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ЛЕСОСИБИРСКИЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ –
филиал Сибирского федерального университета

Высшей математики, информатики, экономики и естествознания
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
Л.Н. Храмова
подпись инициалы, фамилия
«14» 06 2024 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии
код-наименование направления

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ПОСЕТИТЕЛЕЙ КОМПЛЕКСА «БАРХАТ-ПАРК» Г. КРАСНОЯРСКА

Руководитель 14.06.2024 доцент, канд. пед. наук Е.В. Киргизова
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник 14.06.24г А.С. Бондаренко
подпись, дата инициалы, фамилия

Нормоконтролер 14.06.24г. А.В. Фирер
подпись, дата инициалы, фамилия

Лесосибирск 2024

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска» содержит 53 страницы текстового документа, 28 иллюстраций, 3 таблицы, 40 использованных источников.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ANDROID, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, МОБИЛЬНАЯ РАЗРАБОТКА, JAVA.

Цель исследования – теоретически обосновать и разработать мобильное приложение для посетителей комплекса «Бархат-парк» г. Красноярска.

Объект исследования – инструментальные средства разработки мобильного приложения.

Предмет исследования – процесс разработки мобильного

Для достижения поставленной цели необходимо решить следующие задачи исследования:

– на основе анализа учебной и научно-технической литературы по теме исследования определить инструментальные средства разработки мобильных приложений и их характеристики;

– рассмотреть архитектуру и создание пользовательского интерфейса мобильного приложения для комплекса «Бархат-парк»;

– разработать мобильное приложение для комплекса «Бархат-парк».

В ходе написания выпускной квалификационной работы автор принимал участие в конференциях и конкурсах. Так же было принято к публикации 2 статьи.

В результате выполнения выпускной квалификационной работы разработано мобильное приложение для посетителей комплекса «Бархат-парк» г. Красноярска.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Развитие мобильных технологий существенно изменило способы взаимодействия людей с окружающим миром. Мобильные приложения стали неотъемлемой частью повседневной жизни, обеспечивая удобный доступ к информации и услугам в различных сферах, включая развлечения и отдых. Разработка специализированных мобильных приложений для развлекательных комплексов представляет собой важное направление, способствующее улучшению сервиса и повышению удовлетворенности посетителей.

Комплекс «Бархат-Парк» в городе Красноярске является популярным местом для отдыха и развлечений, предлагая широкий спектр услуг для посетителей всех возрастов. «Бархат-Парк» сталкивается с задачей эффективного информирования и обслуживания своих гостей. Традиционные методы, такие как печатные брошюры и информационные стенды, зачастую оказываются недостаточно эффективными в условиях современного цифрового мира.

Цель исследования – теоретически обосновать и разработать мобильное приложение для посетителей комплекса «Бархат-парк» г. Красноярска.

Объект исследования – инструментальные средства разработки мобильного приложения.

Предмет исследования – процесс разработки мобильного приложения.

Для достижения поставленной цели необходимо решить следующие задачи исследования:

– на основе анализа учебной и научно-технической литературы по теме исследования определить инструментальные средства разработки мобильных приложений и их характеристики;

– выбрать архитектуру и создать пользовательский интерфейс мобильного приложения для комплекса «Бархат-парк»;

– разработать мобильное приложение для комплекса «Бархат-парк».

Методы к исследованию:

– теоретические методы включают в себя анализ учебной и научно-технической литературы по теме исследования, обобщение полученных данных и сравнение;

– эмпирические методы исследования включают в себя наблюдение, проведение бесед, моделирование и тестирование.

Результаты исследования представлены на научных мероприятиях:

1. VII Всероссийском научно-практической конференции «Актуальные проблемы преподавания дисциплин естественнонаучного цикла» (г. Лесосибирск, ЛПИ – филиал СФУ, 22 ноября 2023 г.);

2. VIII Всероссийском (IX Регионального) молодежном научном форуме «Российское могущество прирастать будет Сибирью» (г. Лесосибирск, ЛПИ – филиал СФУ, 14 декабря 2023 г.);

3. III Всероссийском молодежном научном форуме «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 9 апреля 2024 г.);

4. Молодежном научном-образовательном фестивале «Ступени» в рамках III Всероссийского молодежного научного форума «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 12 апреля 2024 г., диплом 2 степени);

5. Юбилейном XX Международной научной конференции студентов, аспирантов и молодых ученых «ПРОСПЕКТ СВОВОДНЫЙ - 2024» (г. Лесосибирск, ЛПИ – филиал СФУ, 17 апреля 2024 г.).

По результатам исследования приняты к публикации статьи:

1. Бондаренко, А. С. Разработка мобильного приложения для комплекса «Бархат-парк» г. Красноярска / А. С. Бондаренко // III Всероссийском молодежном научном форуме «Современное педагогическое образование: теоретический и прикладной аспекты» / ЛПИ – филиал СФУ – Лесосибирск.

2. Бондаренко, А. С. Разработка мобильного приложения для комплекса

«Бархат-парк» г. Красноярска / А. С. Бондаренко // Юбилейном XX Международной научной конференции студентов, аспирантов и молодых ученых «ПРОСПЕКТ СВОВОДНЫЙ - 2024» / ЛПИ – филиал СФУ – Лесосибирск.

Структура работы – работа состоит из введения, двух глав, заключения и списка используемых источников, включающий в себя 40 наименований. Результаты работы представлены в 3 таблицах и 28 рисунках. В 3 приложениях представлены листинги модулей приложения. Общий объем работы – 53 печатных листа.

1 Теоретические основы разработки мобильного приложения

1.1 Анализ предметной области

Перед началом разработки мобильного приложения для комплекса «Бархат-парк» необходимо определить его функциональность и предметную область. Ключевым аспектом мобильного приложения является обеспечение комфортного посещения гостей комплекса «Бархат-парк», что включает в себя создание удобных и безопасных условий для отдыха и развлечений, эффективное управление очередями и предоставление высококачественного обслуживания.

Приложение позволит посетителям получить актуальную информацию о режиме работы, перечне аттракционов и бронировании билетов онлайн.

Перед началом разработки мобильного приложения для комплекса «Бархат-парк» стоит определиться с операционной системой Android или IOS для мобильного приложения.

«Android – это операционная система (ОС) для мобильных телефонов, смартфонов, коммуникаторов, планшетных компьютеров, электронных книг, цифровых проигрывателей, наручных часов, нетбуков и смартбуков, основанная на ядре Linux.» [4, с. 28].

Разработка приложения с использованием этой платформы обеспечит его доступность для широкого круга пользователей.

Для разработки Android приложений существует полноценная интегрированная среда разработки (IDE) – Android Studio. «К достоинствам Android Studio, помимо широких возможностей, следует отнести бесплатность, а также доступность на всех основных платформах – Linux, MacOS и Windows, а с недавних пор и на Chrome OS.» [24, с. 98].

Разработка приложения Android имеет несколько преимуществ, которые особенно важны для целевой аудитории, целей автоматизации и комфорта, которые представлены в таблице 1.

Таблица 1 – Преимущества Android

Преимущества Android	Описание
Широкое распространение мобильных устройств Android	Приложение для этой платформы сможет охватить максимальное количество потенциальных посетителей комплекса «Бархат-парк» г. Красноярска
Разнообразие устройств и бюджетных вариантов	Android поддерживает широкий спектр устройств различных производителей, включая как дорогие флагманские смартфоны, так и доступные по цене модели. Это важно, учитывая разнообразие посетителей и их разные финансовые возможности
Гибкость в разработке	Разработка приложений для Android предоставляет разработчикам большую гибкость в создании пользовательского интерфейса и функциональности приложения. Это позволит адаптировать приложение точно под потребности и предпочтения посетителей комплекса «Бархат-парк» г. Красноярска
Интеграция с другими сервисами Google	Android интегрирован с другими сервисами Google, такими как Google Maps, Google Pay и другими. Это обеспечит легкость в использовании различных функций, таких как навигация и онлайн-оплата, что улучшит пользовательский опыт
Поддержка многозадачности	Android предоставляет мощные инструменты для многозадачности, включая возможность работы с несколькими приложениями одновременно. Это повышает эффективность и удобство работы на мобильных устройствах

Разработка мобильного приложения для комплекса «Бархат-Парк» города Красноярска на платформе Android является оптимальным решением благодаря широкой распространенности этой операционной системы и поддержке разнообразных устройств, включая бюджетные модели, что позволит охватить максимальную аудиторию посетителей с различными финансовыми возможностями.

Таким образом, приложение будет предоставлять актуальную информацию о режиме работы, аттракционах и возможностях онлайн-бронирования, обеспечивая комфортное и безопасное посещение комплекса «Бархат-парк». Гибкость разработки на ОС Android позволит адаптировать интерфейс и функциональность под нужды целевой аудитории, а интеграция с сервисами Google улучшит пользовательский опыт, что повысит уровень обслуживания и комфорт для посетителей комплекса «Бархат-парк».

1.2 Классификация мобильных приложений

Мобильные приложения играют важную роль в современном мире, предоставляя пользователям доступ к информации, услугам и развлечениям.

«Мобильное приложение – это программное обеспечение, специально разработанное под конкретную мобильную платформу. Предназначено для использования на смартфонах, планшетах, умных часах и других мобильных устройствах.» [6, с. 23].

«В зависимости от целей, требований и технологий разработки, мобильные приложения можно классифицировать на несколько типов: нативные, гибридные и кроссплатформенные приложения.» [24]. Выделим следующие категории, которые представлены в таблице 2.

Таблица 2 – Виды приложения

Виды приложений	**Описание видов приложений**
Нативные приложения	Разработанные специально для определенной платформы и языка программирования (например, Java или Kotlin для Android, Swift для iOS). Они обычно обеспечивают лучшую производительность и доступ к полному набору функций устройства
Веб-приложения	Работают через веб-браузер на устройстве пользователя. Это веб-сайты, которые выглядят как настоящие приложения, но размещаются не на устройстве пользователя
Гибридные приложения	Приложения, которые разрабатываются с использованием веб-технологий (HTML, CSS, JavaScript), упакованные в контейнер, который может выполняться как нативное приложение. Они могут быть более универсальными, но могут страдать от ограничений производительности
Кроссплатформенные приложения	Приложения, доступные через веб-браузер мобильного устройства. Они не требуют загрузки и установки, но могут иметь ограниченный доступ к функциям устройства

Рассмотрим преимущества и недостатки каждого вида мобильного приложения, представленные в таблице 2.

Самый распространённый выбор мобильного приложения – нативные приложения, разрабатываемые для ОС Android с использованием языка программирования Java или Kotlin, что обеспечивает их высокую производительность и полный доступ к функциям устройства.

Достоинства и недостатки нативных приложений представлены на рисунке 1.

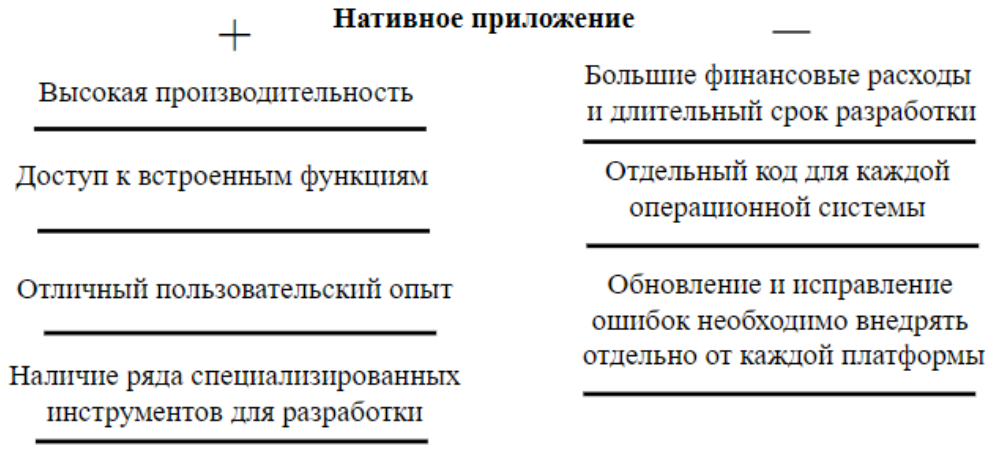

Рисунок 1 – Достоинства и недостатки нативного приложения

Нативные приложения обеспечивают максимальную производительность, доступ к полному набору функций устройства и превосходный пользовательский опыт, что делает их предпочтительным выбором для разработки высококачественных мобильных приложений.

В отличие от традиционных нативных приложений, мобильные веб-приложения работают через веб-браузер и могут использоваться на любом устройстве с доступом в интернет. Это делает их привлекательными как для разработчиков, так и для пользователей, поскольку они не требуют установки.

Достоинства и недостатки веб-приложений представлены на рисунке 2.

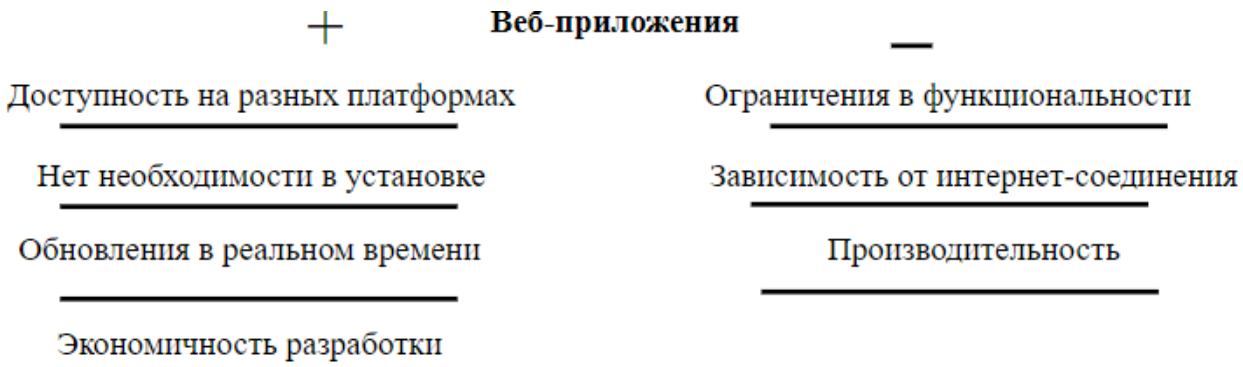

Рисунок 2 – Достоинства и недостатки веб-приложения

Мобильные веб-приложения представляют собой гибкое и экономичное решение для обеспечения доступа к функциональности через веб-браузеры на различных устройствах.

Следующей вид – это гибридные приложения сочетают в себе элементы нативных и веб-приложений. Они работают внутри нативной оболочки, которая отображает веб-контент, написанный с использованием HTML, CSS и JavaScript.

Достоинства и недостатки гибридных приложений представлены на рисунке 3.

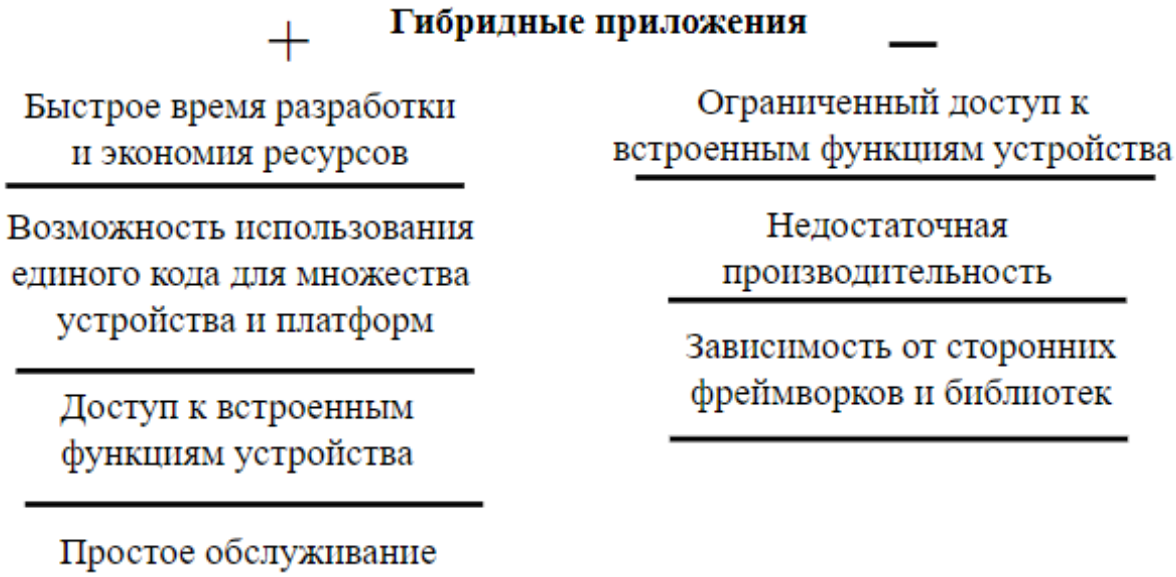

Рисунок 3 – Достоинства и недостатки гибридного приложения

Гибридные приложения предлагают быструю разработку и снижение затрат за счет использования единого кода для разных платформ, хотя и могут уступать нативным в производительности и доступности функций устройства.

Следующие – кроссплатформенные приложения создаются с использованием фреймворков, которые позволяют писать единый код для платформ и затем компилировать его в нативный код для каждой платформы.

Достоинства и недостатки кроссплатформенных приложений представлены на рисунке 4.

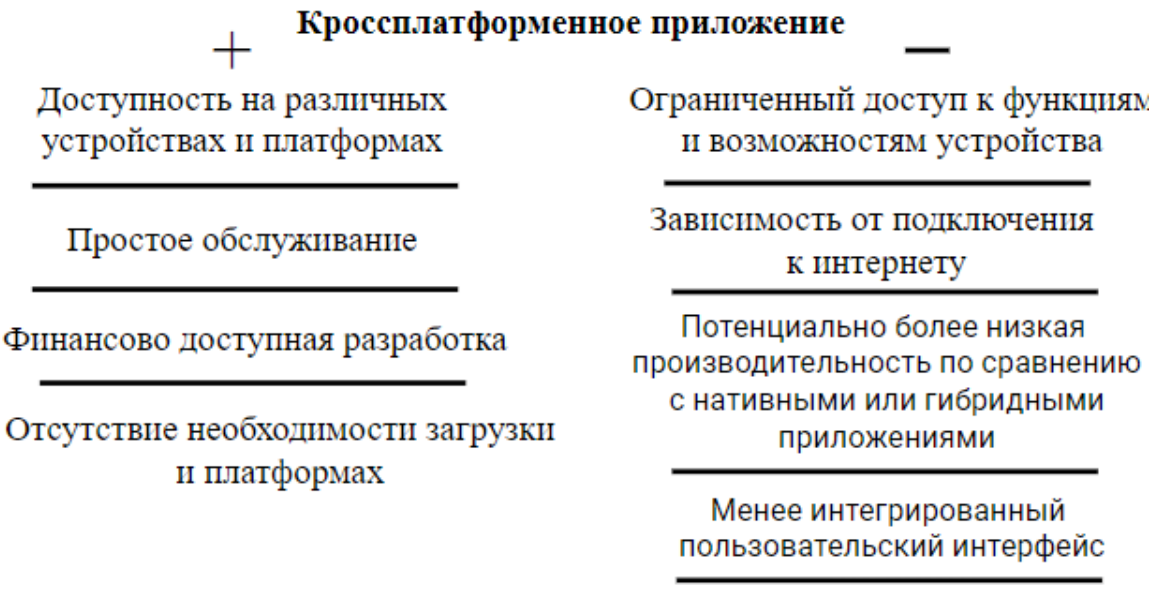

Рисунок 4 – Достоинства и недостатки кроссплатформенного приложения

Кроссплатформенные приложения позволяют использовать единый код для разных платформ, обеспечивая значительную экономию времени и ресурсов при разработке, при этом предлагая близкую к нативной производительность.

После тщательного анализа, выбор был сделан в пользу нативного приложения, потому что оно обеспечивает высочайшую производительность и полный доступ к аппаратным ресурсам и функциям устройства.

Нативные приложения разрабатываются специально для определенной платформы, что позволяет максимально оптимизировать их работу и использовать все преимущества и функциональные возможности этой платформы.

Таким образом, этот подход обеспечивает не только более надежную и безопасную разработку, но и лучший пользовательский опыт благодаря интуитивному интерфейсу и соответствию стандартам дизайна платформы.

Кроме того, нативные приложения лучше адаптируются под разные размеры экранов и разрешения устройств, что делает их внешний вид более привлекательным и удобным для всех пользователей.

1.3 Архитектура мобильного приложения

«Архитектура мобильного приложения – это совокупность правил, методов и шаблонов разработки мобильных приложений.» [7, с. 43].

В Android приложениях используется архитектура Clean для создания гибкого, тестируемого и поддерживаемого кода. «Чистая архитектура разделяет приложение на слои с четкими границами и зависимостями, направленными внутрь, что позволяет легко изменять и расширять систему без влияния на остальные компоненты.» [31].

Это достигается путем разделения кода на уровне с четкими границами и внутренними зависимостями, что облегчает изменение и расширение системы.

Принципы чистой архитектуры также помогают избежать проблем с зависимостями и разделить приложение на логические блоки. Это повышает удобство добавления новых функций и изменения уже существующих. Основные принципы чистой архитектуры, представлены на рисунке 5.

Рисунок 5 – Основные принципы чистой архитектуры

Рассмотрим основные уровни (слои) чистой архитектуры:

– сущности: это внутренний и самый важный слой. Сущности представляют собой бизнес-объекты и инкапсулируют основные бизнес-правила и логику предприятия. Они независимы от конкретных приложений и могут быть использованы повторно в разных проектах;

– сценарии: данный слой определяет конкретные бизнес-правила и сценарии использования приложения. Он описывает, как сущности взаимодействуют друг с другом и какие операции могут выполняться. Сценарии зависят от сущностей, но независимы от внешних интерфейсов;

– интерфейс-адаптеры: этот слой отвечает за преобразование данных между внутренними слоями (сущности и сценарии) и внешними слоями (фреймворки и драйверы);

– фреймворки и драйверы: «это самый внешний слой, который включает в себя детали реализации, такие как пользовательский интерфейс (UI), база данных, внешние API, устройства и веб. Этот слой зависит от внутренних слоев, но внутренние слои не зависят от него.» [11].

Каждый из этих слоев имеет четко определенные обязанности и взаимодействует с другими слоями через интерфейсы. Основной принцип чистой архитектуры – зависимость только внутрь, то есть внутренние слои не знают о существовании внешних.

Это обеспечивает модульность, тестируемость и устойчивость к изменениям в коде. В сравнении с другими подходами, Clean Architecture предоставляет более явное и понятное разделение приложения на компоненты. Разработчик может легко видеть, как каждая часть приложения взаимодействует друг с другом, что упрощает сопровождение и устранение ошибок.

«Чистая архитектура является мощным подходом к построению масштабируемых систем, обеспечивая независимость и гибкость кода» [15, с. 153], схема чистой архитектуры представлена на рисунке 6.

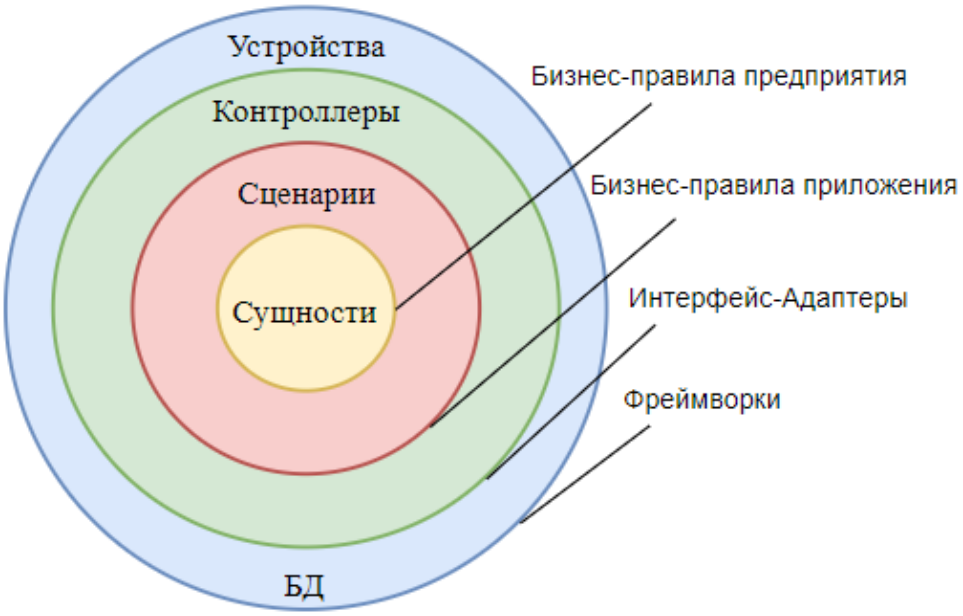

Рисунок 6 – Схема чистой архитектуры мобильного приложения

Чистая архитектура обычно представляется в виде круга из четырех слоев. Внутренний слой, «Сущности», содержит бизнес-правила предприятия, используемую во многих приложениях. Последовательно следуют «Сценарии», которые содержит бизнес-правила приложения. Следующий круг содержит информацию об интерфейс-адаптерах. И самый внешний слой, фреймворки и драйверы, включающий UI, базу данных, внешние интерфейсы устройства.

Таким образом, четкая структура и независимость слоев, обеспечивает разработчику мобильного приложения простоту в тестировании и устранении неполадок, независимость от пользовательского интерфейса, баз данных и внешних фреймворков, а также удобной для установки различных плагинов.

1.4 Моделирование бизнес-процессов

«Моделирование бизнес-процессов является важнейшим элементом разработки программного обеспечения, так как оно позволяет детально изучить и описать текущие операции и рабочие потоки внутри организации.» [14].

Моделирование бизнес-процессов включает в себя использование различных методов и инструментов для представления и анализа текущих и

предполагаемых процессов. Основные методы моделирования представлены на рисунке 7.

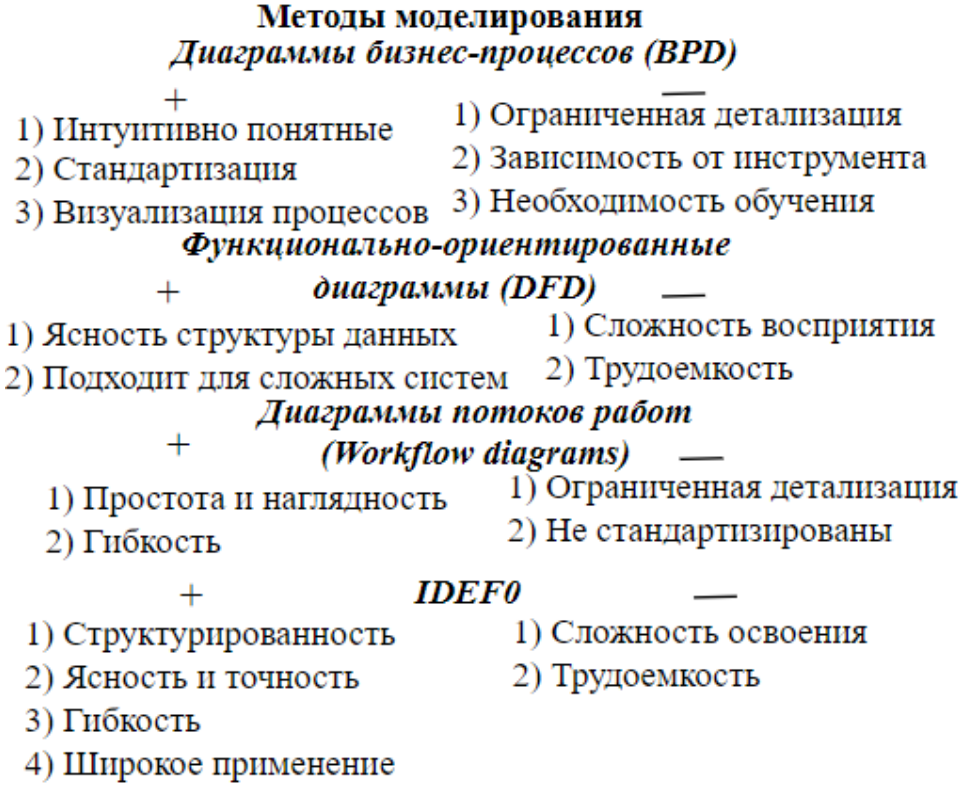

Рисунок 7 – Методы моделирования

Изучив и сравнив все методы, можно сделать вывод, что IDEF0 является наиболее подходящим инструментом для моделирования бизнес-процессов, потому что он предоставляет структурированный и формализованный подход к описанию функциональных аспектов, позволяя детализировать взаимосвязи между элементами процесса и их влияние на общую систему.

«IDEF0 (Integration Definition for Function Modeling) – это методология для функционального моделирования, которая используется для описания процессов, функций и их взаимодействий в системе.» [3].

Диаграмма первого уровня представлена в формате IDEF0 и описывает основные функции приложения для аквапарка. В диаграмме, изображенной на рисунке 8, представлены входные данные и выходные данные приложения.

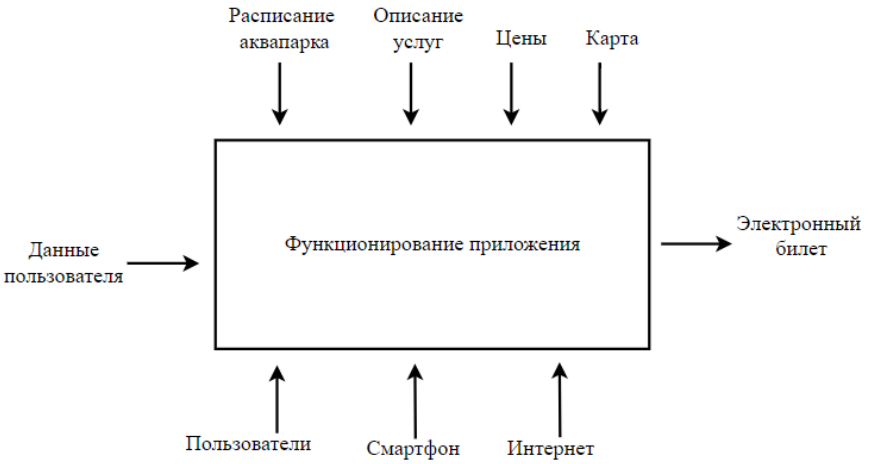

Рисунок 8 – Диаграмма первого уровня IDEF0

«Диаграмма первого уровня IDEF0 отображает ясное понимание структуры и работы приложения, что важно для дальнейшего анализа и оптимизации процессов.» [23].

Диаграмма второго уровня представляет детализированный процесс создания приложения, включающий пять основных функции: редактирование данных пользователя, редактирование расписание аквапарка, редактирование описания услуг, редактирование описания цен и редактирование карты аквапарка. Выходными данными будет являться окончательное приложение.

Каждая функция взаимосвязана и последовательно передает данные следующему этапу. Диаграмма второго уровня IDEF0 представлена на рисунке 9.

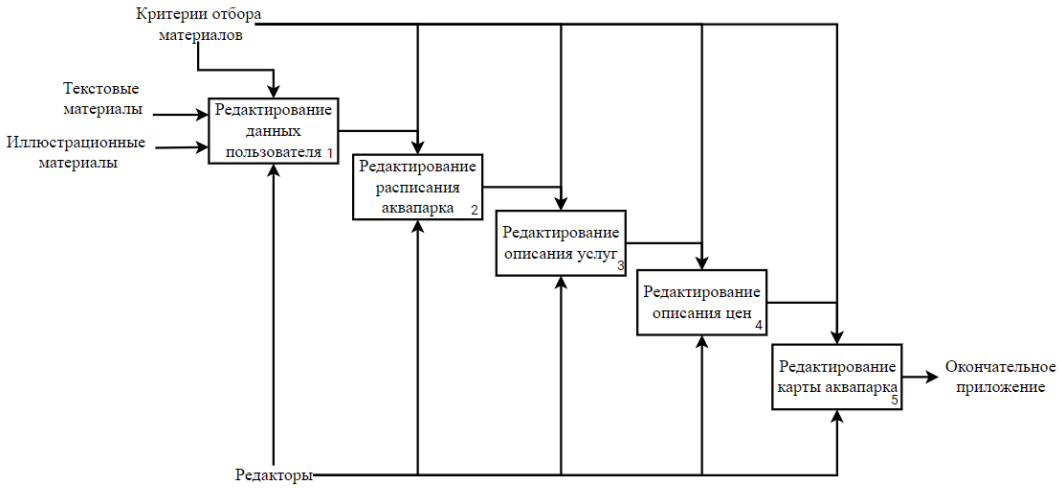

Рисунок 9 – Диаграмма второго уровня IDEF0

Эта диаграмма наглядно демонстрирует последовательность и взаимосвязь этапов процесса создания приложения, обеспечивая четкое понимание всех шагов и их результатов.

Таким образом, моделирование бизнес-процессов с использованием методологии IDEF0 представляет собой эффективный подход к анализу и оптимизации. Диаграммы первого, второго уровня IDEF0 обеспечивают ясное представление о структуре и последовательности действий в процессе создания и редактирования, что позволяет эффективно управлять и контролировать каждый этап разработки.

2 Разработка мобильного приложения

2.1 Разработка дизайна мобильного приложения

Разработка интерфейса пользователя играет ключевую роль в успешности любого приложения. Удобный, интуитивный и привлекательный интерфейс значительно улучшает пользовательский опыт, повышает удовлетворенность и удержание пользователей.

На рисунке 10 рассматриваются основные принципы разработки интерфейса, которые необходимо учитывать при создании мобильных приложений. Эти принципы помогают обеспечить высокое качество взаимодействия с пользователями и создать приложение, соответствующий их ожиданиям и потребностям.

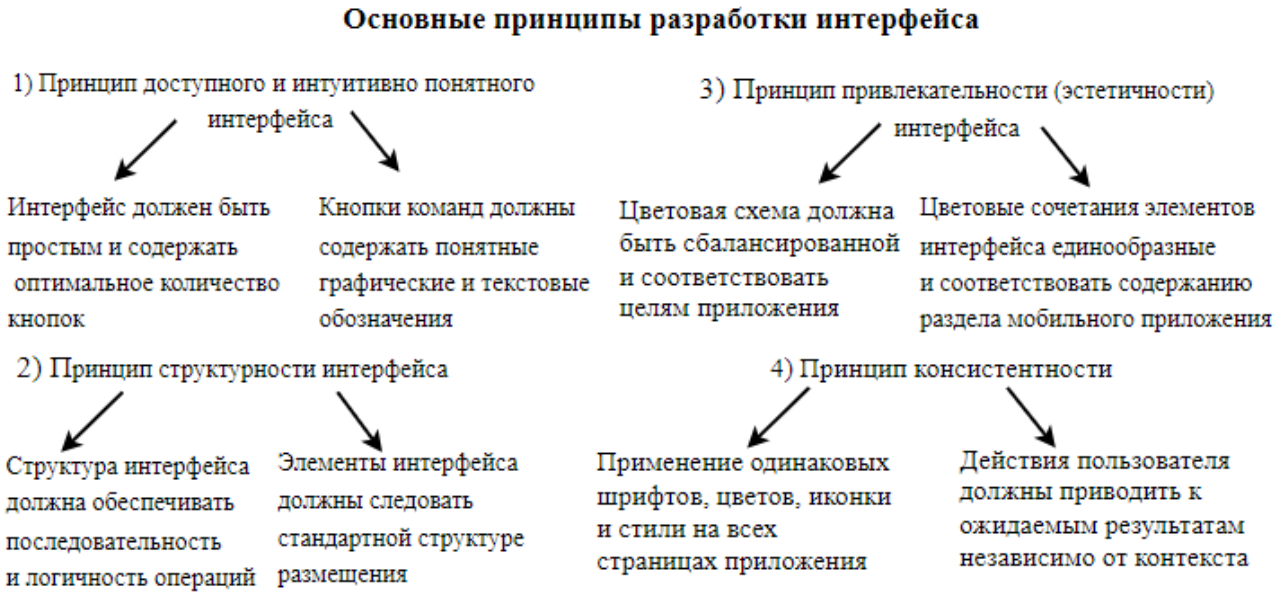

Рисунок 10 – Основные принципы разработки интерфейса

Следование этим принципам при разработке интерфейса позволяет создать приложения, которые не только удовлетворяют функциональные требования, но и обеспечивают положительный пользовательский опыт.

Разработка дизайна мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска была разделена на три этапа. Дизайн мобильного

приложения был разработан в онлайн-сервисе «Miro» – это бесплатный сервис для проектирования внешнего вида мобильных приложений.

Первым этапом разработки дизайна мобильного приложения является создание прототипа мобильного приложения.

«Прототип – это интерактивная модель приложения, которая демонстрирует его функциональность и навигацию без деталей дизайна.» [16]. На этапе прототипа определяются основные функции и возможности приложения, которые будут включены в прототип.

Прототип мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска представлен на рисунке 11.

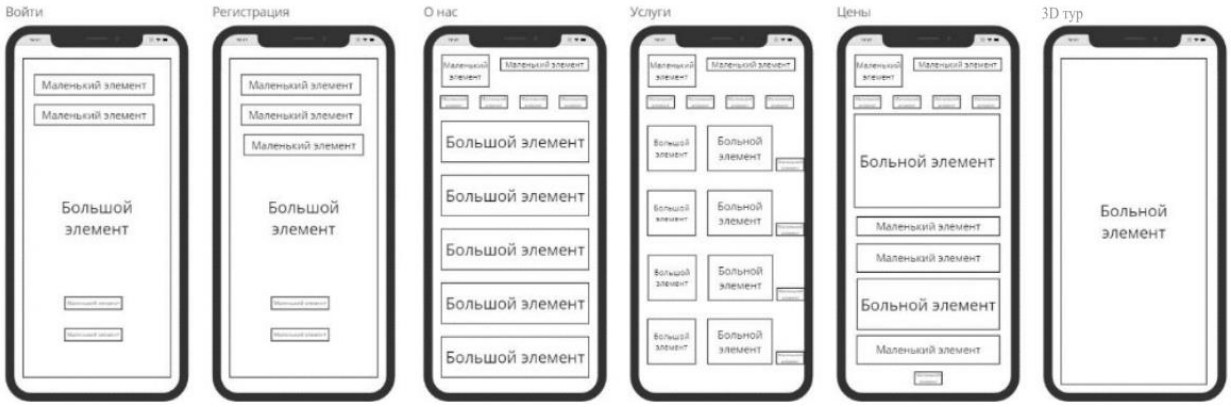

Рисунок 11 – Прототип мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска

Вторым этапом разработки дизайна мобильного приложения для посетителей комплекса «Бархат-парк» будет создание вайрфрейма.

«Вайрфрейм – это набор черновых макетов экранов приложения, который показывает расположение элементов интерфейса и их взаимосвязь, но без деталей дизайна.» [18, с. 43]. На этапе создания вайрфрейма определяется структура приложения и его основные разделы, а также контент, который будет представлен на каждом экране.

Вайрфрейм мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска представлен на рисунке 12.

Рисунок 12 – Вайрфрейм мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска

Третьим и последним этапом является финальное оформление макета мобильного приложения для посетителей комплекса «Бархат-парк».

На этом этапе макет мобильного приложения приобретает свой окончательный внешний вид с учетом деталей дизайна. Происходит выбор основных цветов и шрифтов, которые будут использоваться в приложении, чтобы создать единый стиль мобильного приложения.

Финальное оформление макета мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска представлено на рисунке 13.

Рисунок 13 – Финальное оформление макета мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска

В процессе разработке дизайна мобильного приложения для комплекса «Бархат-парк» важными этапами являются создание прототипа, разработка вайрфрейма и финальное оформление макета.

Прототип демонстрирует функциональность и навигацию без углубления в детали дизайна, тогда как вайрфрейм определяет структуру приложения и расположение элементов интерфейса.

Финальное оформление макета добавляет детали дизайна, цветовую схему и графические элементы, создавая законченный внешний вид.

Тщательное выполнение каждого этапа обеспечивает разработку приложения, отвечающего потребностям и ожиданиям пользователей комплекса «Бархат-парк» г. Красноярска.

2.2 Выбор средств разработки

В процессе создания мобильного приложения важно выбрать подходящие средства разработки, которые обеспечат эффективность работы, высокое качество и удовлетворение требований проекта.

В данной главе рассматриваются ключевые критерии, влияющие на выбор среды разработки, а также проводится обзор и сравнение наиболее популярных инструментов, используемых в современной практике.

Это поможет определить наиболее подходящие решения для реализации конкретных проектов и обеспечения высокой эффективности разработки.

Рассмотрим несколько возможных сред разработки.

1. «Android Studio — это официальная интегрированная среда разработки (IDE) для создания приложений на платформе Android. Разработанная компанией Google, она предлагает полный набор инструментов для разработки, тестирования и отладки приложений для Android.» [24].

Среда разработки «Android Studio» представлена на рисунке 14.

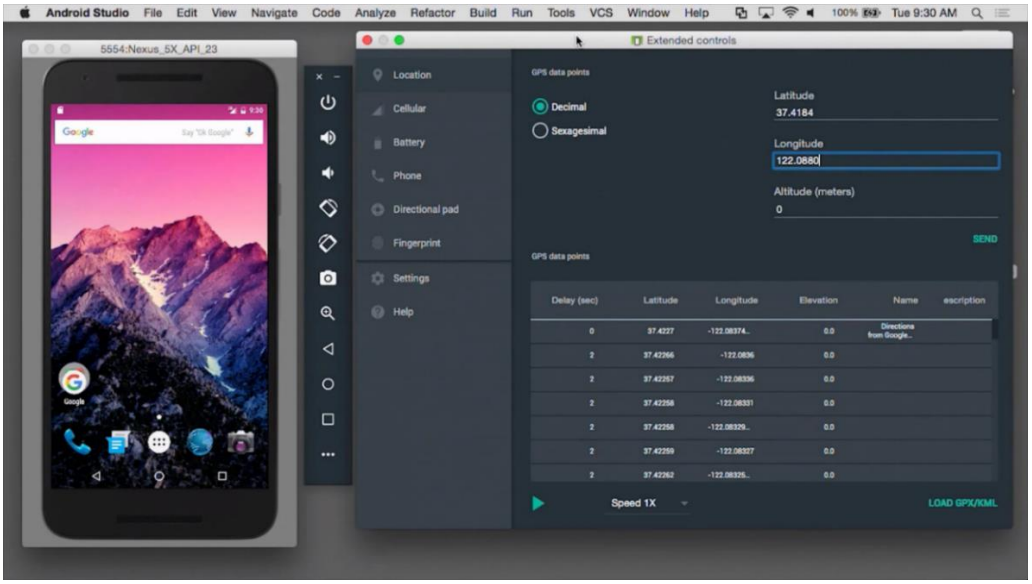

Рисунок 14 – Среда разработки «Android Studio»

Android Studio обладает многими преимуществами, она также имеет некоторые недостатки, которые могут оказывать влияние на процесс разработки. Рассмотрим подробнее основные качества Android Studio.

Преимущества и недостатки Android Studio представлены на рисунке 15.

Рисунок 15 – Преимущества и недостатки Android Studio

Факторы, предоставленные в схеме следует учитывать при выборе среды разработки для конкретного проекта, чтобы обеспечить оптимальные условия для создания мобильных приложений.

2. «Xamarin – это платформа для разработки кроссплатформенных мобильных приложений с использованием языка программирования C#. Она

позволяет создавать нативные приложения для Android, iOS и Windows.» [40].

Среда разработки «Xamarin» представлена на рисунке 16.

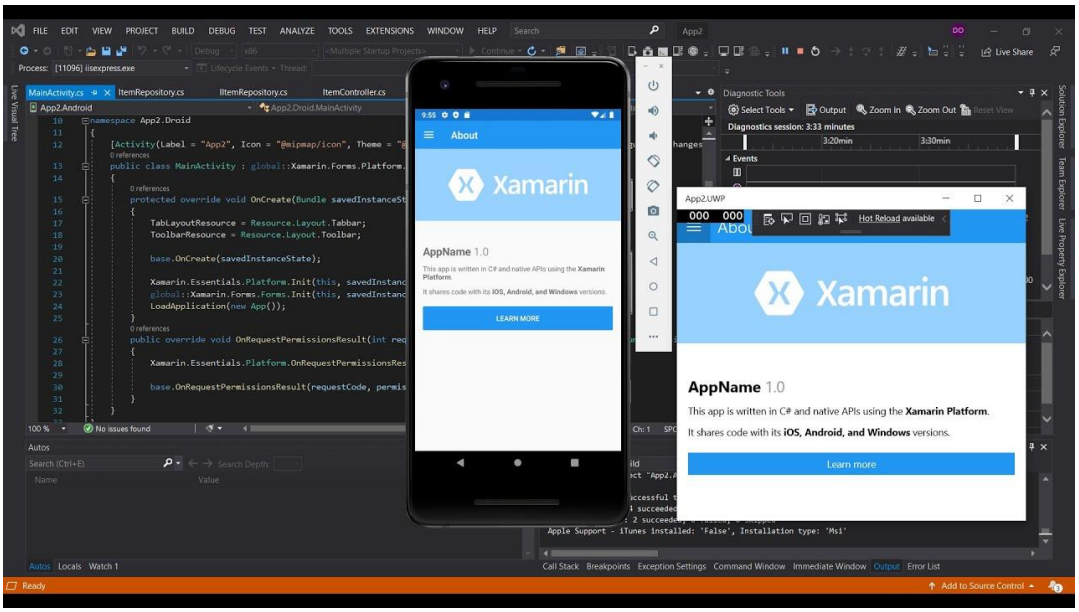

Рисунок 16 – Среда разработки «Xamarin»

Xamarin является одной из популярных сред для создания мобильных приложений. Кроссплатформенная среда разработки, позволяющая создавать приложения для различных платформ. Выделим основные преимущества и недостатки Xamarin, которые представлены на рисунке 17.

Рисунок 17 – Преимущества и недостатки «Xamarin»

Эти аспекты следует принимать во внимание при выборе Xamarin в качестве инструмента для разработки.

3. «Appcelerator Titanium – это кроссплатформенная среда разработки приложений, которая позволяет разработчикам создавать нативные мобильные приложения для платформ Android и iOS.» [29].

Среда разработки «Appcelerator Titanium» представлена на рисунке 18.

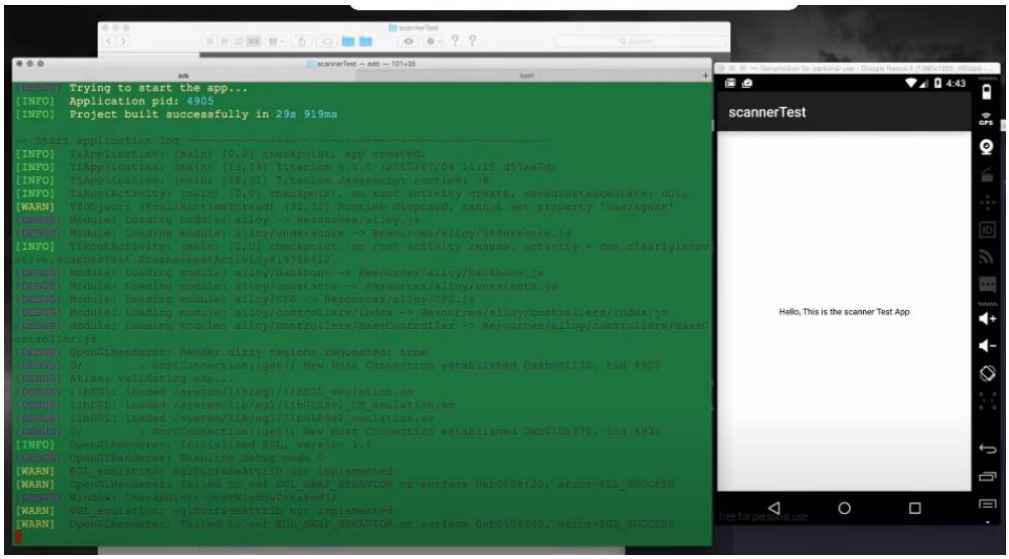

Рисунок 18 – Среда разработки «Appcelerator Titanium»

Современные технологии разработки мобильных приложений предоставляют разработчикам множество инструментов для создания приложений. Одним из инструментов является Appcelerator Titanium, разберем подробно все преимущества и недостатки, представленных на рисунке 19.

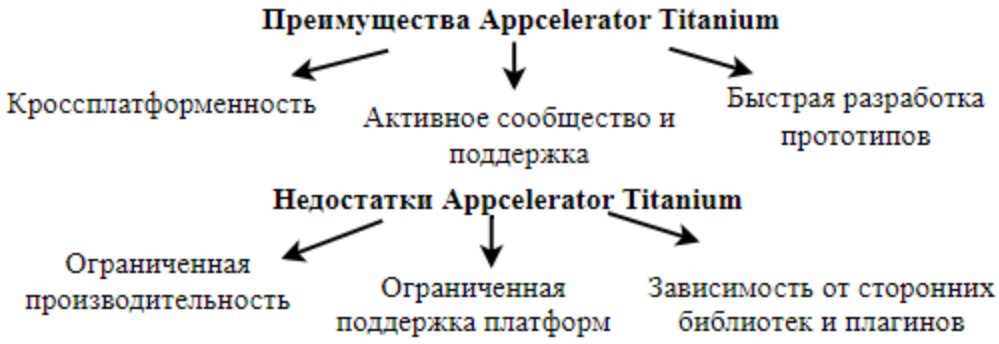

Рисунок 19 – Преимущества и недостатки «Appcelerator Titanium»

Несмотря на все свои преимущества, у платформы также есть некоторые ограничения, включая ограниченную производительность и зависимость от сторонних библиотек и плагинов.

Для разработки мобильного приложения для комплекса «Бархат-парк» г. Красноярска была выбрана среда разработки «Android Studio». Стоит отметить, что Android Studio – универсальная среда разработки, так как позволяет оптимизировать работу будущих приложения для работы не только на смартфонах, но и так же на планшетах и портативных персональных компьютерах.

Выбор остановился на Android Studio по нескольким основным причинам: небольшой опыт в написании мобильных приложений на языках Java/Kotlin, гибкость среды разработки и поддержка компанией Google.

Таким образом, Android Studio представляет собой идеальное средство для разработки мобильного приложения для посетителей комплекса «Бархат-парк» г. Красноярска, учитывая его преимущества и возможности, а также ограниченный опыт в мобильной разработке и поддержку со стороны компании Google.

2.3 Компоненты Android приложения

Разработка приложений для операционной системы Android требует понимания основных строительных блоков, из которых состоят приложения. В Android-приложениях все компоненты играют важную роль в обеспечении функциональности, взаимодействия с пользователем и интеграции с другими частями системы. Понимание этих компонентов и их правильного использования является одним из ключевых аспектов успешного создания Android-приложений.

В Android существуют четыре типа компонентов: Activities, Services, Broadcast receivers и Content providers.

Эти компоненты представляют собой строительные блоки, из которых состоят приложения, и каждый из них выполняет свою уникальную функцию, компоненты представлены на рисунке 20.

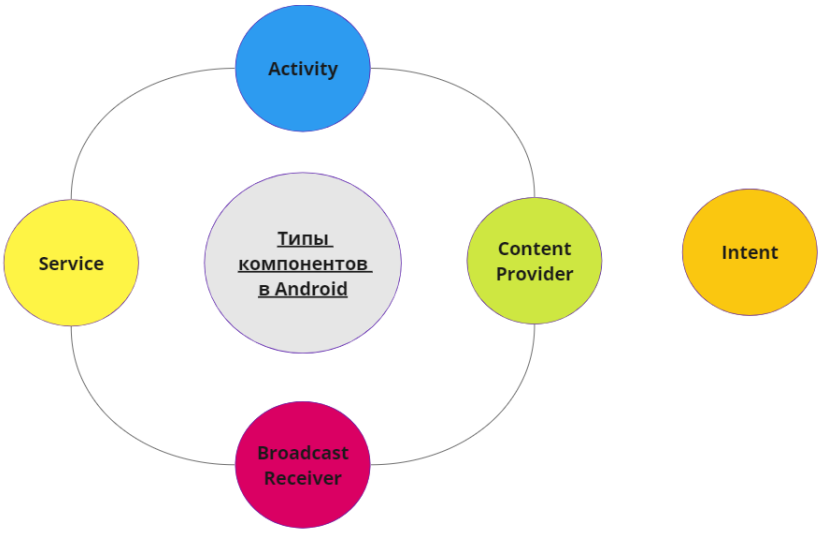

Рисунок 20 – Схема типов компонентов в Android

Объекты Intents играют важную роль в Android приложениях, поскольку почти все действия осуществляются через них. Intent представляет собой механизм для описания одного действия, такого как выбор фотографии, отправка сообщения, совершение звонка или открытие браузера.

«Самый распространенный сценарий использования Intent – запуск другого Activity в рамках текущего приложения.» [6, с. 54].

Activity представляет собой пользовательский интерфейс для определенного действия, которое пользователь может совершить. Например, приложение для обмена сообщениями может иметь Activity для просмотра списка контактов, для написания сообщения и для просмотра полученных сообщений. Activity может находиться в одном из трех состояний: активном (running), приостановленном (paused) и остановленном (stopped). Переход между состояниями уведомляет об этом систему, вызывая соответствующие методы, такие как onCreate(), onStart(), onResume(), onPause(), onStop() и onDestroy(), которые представлены в таблице 3.

Методы жизненного цикла Activity позволяют управлять взаимодействием пользователя с интерфейсом и обеспечивают плавный переход между состояниями приложения.

Таблица 3 – Методы Activity

Название методов Activity	**Описание методов Activity**
onCreate()	Метод вызывается системой при создании Activity. Здесь происходит инициализация Activity, например, установка макета пользовательского интерфейса и выполнение других необходимых настроек. Обычно этот метод используется для одноразовой инициализации, которая должна произойти только один раз во время жизненного цикла Activity
onStart()	Этот метод вызывается системой, когда Activity становится видимым для пользователя, но еще не начала взаимодействие с пользователем. Например, после того как Activity было создано методом onCreate(), но перед тем, как пользователь начал взаимодействие с Activity
onResume()	Метод вызывается системой, когда Activity становится активным и готовым к взаимодействию с пользователем. В этом методе обычно выполняются операции, которые необходимо проводить при каждом входе в Activity, такие как запуск анимаций или запуск потоков
onPause()	Метод вызывается системой, когда Activity теряет фокус, но остается видимым для пользователя. Например, когда другое Activity становится на передний план, но текущее Activity остается видимым в частично перекрытом состоянии. В этом методе обычно производят сохранение состояния Activity и приостановку выполнения операций, которые могут быть ресурсоемкими
onStop()	Метод вызывается системой, когда Activity больше не видимо пользователю. Это происходит, когда другое Activity полностью перекрывает текущее Activity или когда текущее Activity завершается. В этом методе обычно производится освобождение ресурсов и подготовка к завершению работы Activity
onDestroy()	Метод вызывается системой, когда Activity завершает свою работу и будет уничтожено. Это может произойти по причине завершения работы приложения или из-за вызова метода finish() в самом Activity. В этом методе обычно производится освобождение всех ресурсов, связанных с Activity, и выполнение финальных операций перед его уничтожением

Жизненный цикл activity в разработке мобильных приложений для Android представляет собой последовательность состояний, через которые проходит activity, начиная с её создания и заканчивая уничтожением.

Понимание этого цикла важно для правильного управления состоянием приложения и обеспечения его стабильной работы. В рамках данной схемы рассматриваются ключевые этапы жизненного цикла activity, включая создание, запуск, приостановку, возобновление и завершение, что позволяет разработчикам эффективно управлять ресурсами и отвечать на взаимодействия пользователя. Жизненный цикл Activity представлен на рисунке 21.

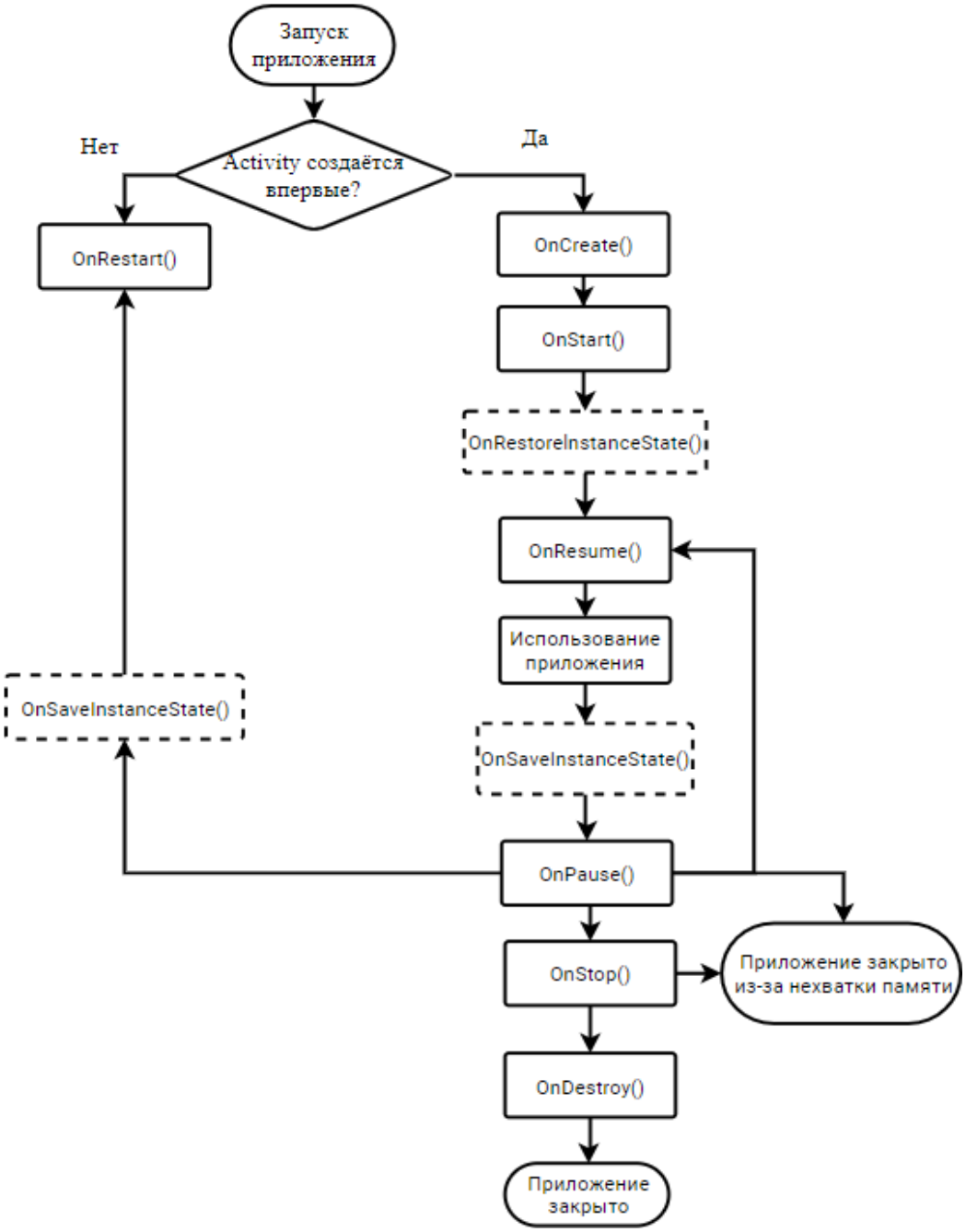

Рисунок 21 – Жизненный цикл Activity

Использование схемы жизненного цикла activity позволяет структурировать код и оптимизировать взаимодействие с системой, что в конечном итоге способствует созданию более стабильного и эффективного приложения.

Другой важный компонент Android приложений – Service. В отличие от Activity, Service не имеет пользовательского интерфейса и предназначен для выполнения фоновых задач. «Service – это некий процесс, который запускается в фоновом режиме.» [21, с. 72]. Как пример, Service может получать данные по сети, выполнять какие-либо длительные вычисления. Хорошим примером Service служит проигрыватель музыки.

Как и Activities, Services запускаются в главном потоке процесса приложения. По этой причине их следует запускать в отдельном потоке, чтобы они не блокировали другие компоненты или пользовательский интерфейс.

Broadcast receiver – это компонент, который ничего не делает, кроме того, что рассылает и реагирует на широковещательные сообщения. Примером широковещательных компонентов могут быть: сообщения об переходе на летнее/зимнее время, сообщения об минимальном заряде батареи и так далее.

Broadcast receiver не отображает пользовательский интерфейс, но может запустить Activity на полученное сообщение или использовать NotificationMana ger для привлечения внимания пользователя. Привлечь внимание пользователя можно, вибрацией устройства, проигрыванием звука или миганием вспышки.

Приемник широковещательных сообщений имеет единственный метод жизненного цикла: onReceive(). Когда широковещательное сообщение прибывает для получателя, Android вызывает его методом onReceive() и передает в него объект Intent, содержащий сообщение. Приемник широковещательных сообщений является активным только во время выполнения этого метода.

Content providers предоставляют доступ к данным (чтение, добавление, обновление). Content provider может предоставлять доступ к данным не только своему приложению, но и другим.

Из вышеизложенного следует, что эффективное использование этих методов позволяет разработчикам создавать приложения, которые реагируют на действия пользователя и оптимизируют использование системных ресурсов.

2.5 Руководство пользователя

Руководство пользователя, предназначено для облегчения работы с приложением, включает подробные инструкции по использованию основных функций, описание объяснение ключевых компонентов системы.

В руководстве пользователя структурная схема играет важную роль, поскольку она предоставляет визуальное представление различных компонентов и их взаимосвязей в системе или приложении. Структурная схема представлена на рисунке 22.

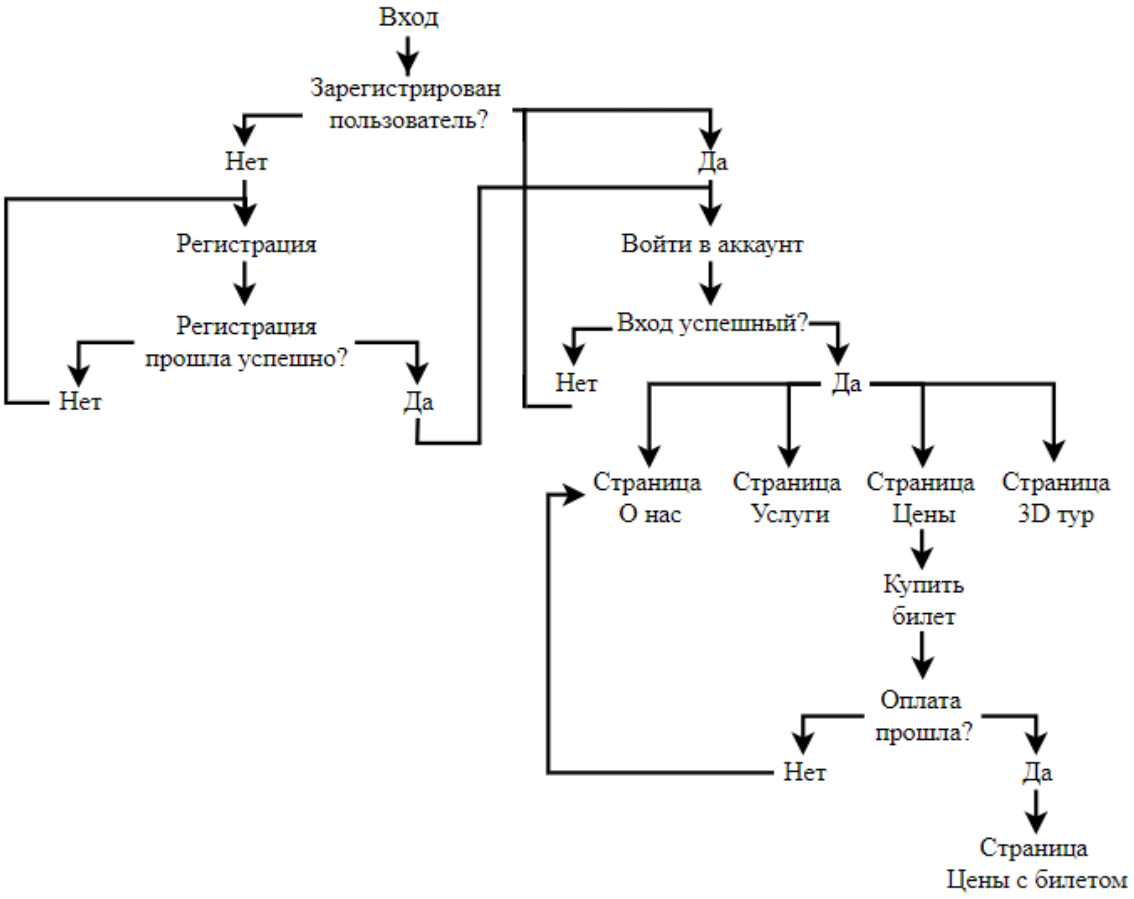

Рисунок 22 – Структурная схема приложения

Структурная схема помогает пользователям быстро ориентироваться в приложении, улучшая их понимание и повышая эффективность работы с приложением.

Первым, что должен сделать пользователь для начала работы с приложением, является загрузка и установка apk-файла на свое мобильное устройство. Важно убедиться, что версия операционной системы вашего устройства составляет Android 5.0 или более позднюю.

После завершения установки появится иконка приложения на экране телефона. Чтобы запустить приложение, нужно нажать на эту иконку. Приложение откроется, и вы сможете начать его использование.

Если вы столкнетесь с какими-либо проблемами при установке или запуске, убедитесь, что ваше устройство соответствует всем требованиям, а также попробуйте перезагрузить телефон и повторить попытку.

Работа с приложением:

Стартовой страницей является регистрация пользователя, где вам потребуется внести данные, такие как: логин и пароль. Если произойдет ошибка в совпадениях пароля, при нажатии на кнопку регистрации, система выдаст сообщение, что пароли не совпадают. Страница регистрации мобильного приложения для комплекса «Бархат-парк» продемонстрирована на рисунке 23.

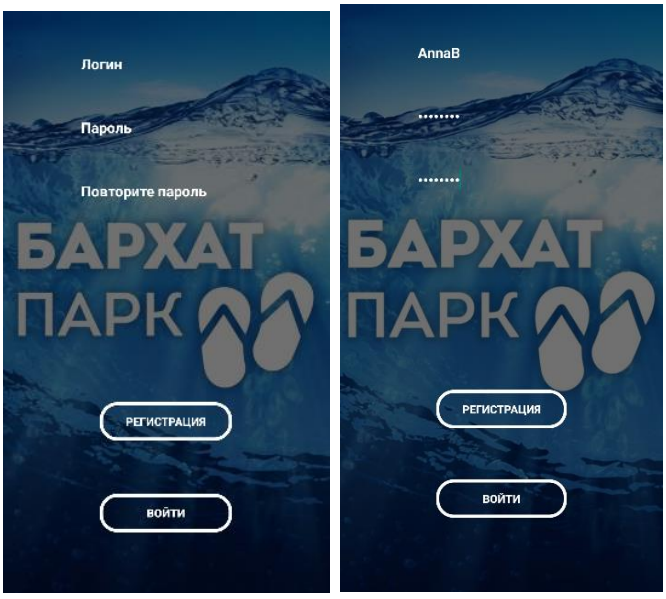

Рисунок 23 – Страница регистрации

После успешной регистрации вы перейдете на страницу авторизации, где необходимо будет повторно ввести свои учетные данные. Если введенные данные не совпадут с теми, что были указаны при регистрации, система уведомит вас об этом с помощью сообщения и предложит повторить попытку ввода данных.

Таким образом, приложение обеспечит надежную проверку и защиту ваших данных, предотвращая несанкционированный доступ и обеспечивая высокий уровень безопасности пользовательской информации.

Страница авторизации мобильного приложения для комплекса «Бархат-парк» представлена на рисунке 24.

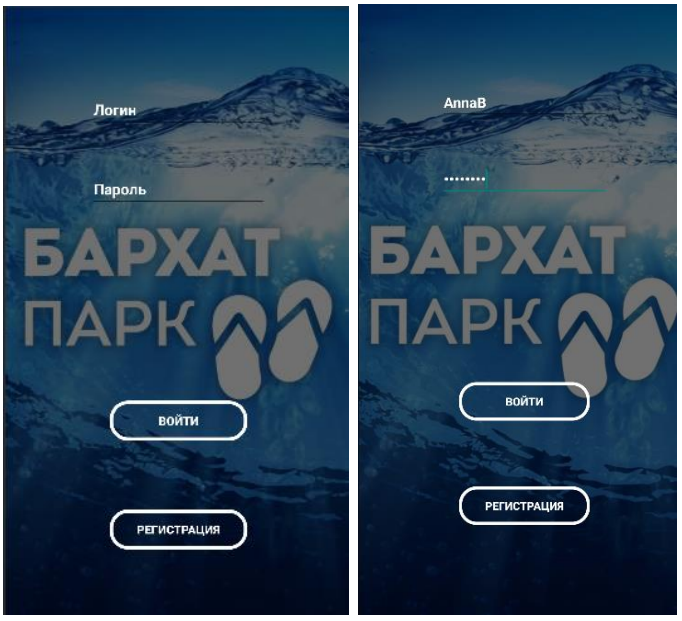

Рисунок 24 – Страница авторизации

После успешной авторизации пользователь попадает на главную страницу мобильного приложения, где находится основная информация о комплексе «Бархат-парк», дальше пользователь может перейти на три дополнительных страницы, такие как: услуги, цены и 3D тур.

Главная страница мобильного приложения для комплекса «Бархат-парк» представлена на рисунке 25.

Рисунок 25 – Главная страница приложения

Одним из ключевых разделов является страница услуг. На этой странице пользователи могут просмотреть полный список доступных услуг комплекса «Бархат-парк», описания каждой из них. Благодаря удобной навигации и четкому оформлению, пользователи могут быстро найти интересующую информацию об услуге. Страница «Услуги» мобильного приложения для посетителей комплекса «Бархат-парк» представлена на рисунке 26.

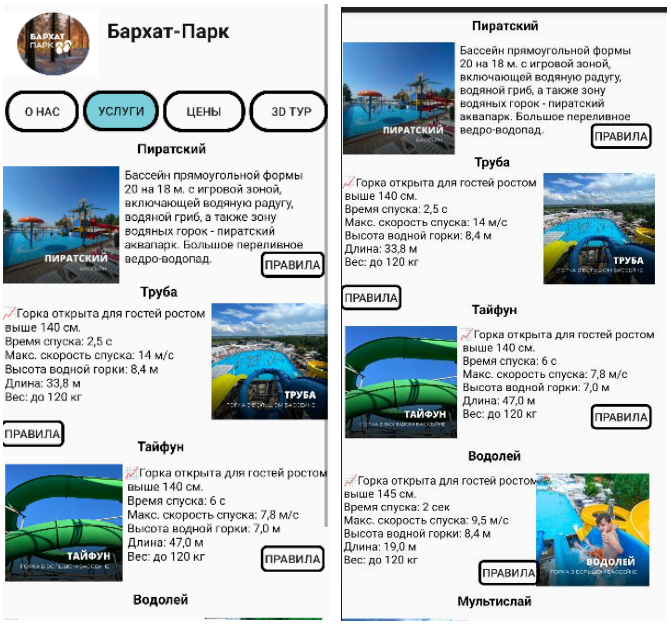

Рисунок 26 – Страница «Услуги»

На следующей странице пользователь имеет возможность подробно ознакомиться с актуальными ценами на различные билеты, что позволит ему сделать обоснованный выбор. После этого, используя удобный интерфейс, он может выбрать наиболее подходящую для себя дату и время мероприятия, учитывая свои личные предпочтения и расписание.

Следуя простым инструкциям, пользователь может перейти к процессу оформления покупки, выполняя все необходимые шаги для выбора нужного билета, ввода личных данных и осуществления платежа. Страница «Цены» мобильного приложения для посетителей комплекса «Бархат-парк», которая предоставляет всю необходимую информацию о доступных билетах и их стоимости, представлена на рисунке 27.

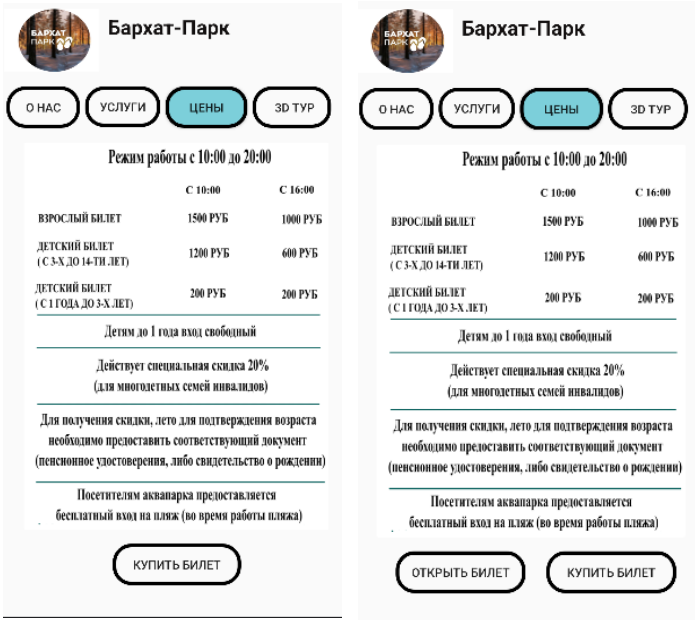

Рисунок 27 – Страница «Цены»

Затем пользователь увидит интерактивную виртуальную карту комплекса «Бархат-парк», на которой будут отображаться различные аттракционы, бассейны и зоны отдыха, а также предоставлены инструменты для перемещения и поиска интересующих объектов. Страница «3D тур» мобильного приложения для посетителей комплекса «Бархат-парк» представлена на рисунке 28.

Рисунок 28 – Страница «3D тур»

В заключении данной главы, представленной в приложении, пользователь получает всестороннее представление о функциональности и возможностях приложения. От установки до работы с основными функциями, каждый шаг описан подробно и доступно.

Пользователю предоставляется информация о регистрации, авторизации, а также основные разделы приложения, такие как услуги, цены и 3D тур аквапарка. Благодаря наглядным иллюстрациям, четким описаниям и удобному интерфейсу, пользователь может легко ориентироваться и использовать приложение с уверенностью.

Интуитивно понятный интерфейс и логичная организация информации обеспечивают эффективное взаимодействие с приложением. Благодаря этому пользователи могут быстро и легко находить нужные сведения и функции.

Таким образом, созданное руководство пользователя является важным инструментом для обеспечения качественного и комфортного пользовательского опыта. Подробное и доступное руководство пользователя способствует долгосрочной привязанности клиентов к приложению, что является важным фактором для успешного функционирования и развития комплекса в условиях высокой конкуренции на рынке.

ЗАКЛЮЧЕНИЕ

В рамках данного дипломного проекта были рассмотрены различные аспекты разработки программного обеспечения, включая выбор среды разработки, моделирование бизнес-процессов и создание руководства пользователя. Была проведена аналитика существующих методов и инструментов. Полученное руководство пользователя является важной частью проекта, обеспечивая пользователям всю необходимую информацию для эффективного использования разработанного приложения.

Для разработки приложения под Android был выбран оптимальный для данного случая набор инструментов, в который вошли: язык программирования Java, Google App Script, база данных Google Sheets, а также сторонние библиотеки для Android.

Во время разработки мобильного приложения были изучены основы языка программирования Java, подробно изучены скрипты Google, а также взаимодействие клиента приложения с Android API.

Так же были выделены функциональные требования к мобильному приложению.

В ходе написания выпускной квалификационной работы были участия в конференциях и конкурсах. Так же опубликовано несколько статей, представленных на следующих конференциях и конкурсах:

1. VII Всероссийском научно-практической конференции «Актуальные проблемы преподавания дисциплин естественнонаучного цикла» (г. Лесосибирск, ЛПИ – филиал СФУ, 22 ноября 2023 г., участие);

2. VIII Всероссийском (IX Регионального) молодежном научном форуме «Российское могущество прирастать будет Сибирью» (г. Лесосибирск, ЛПИ – филиал СФУ, 14 декабря 2023 г., участие);

3. III Всероссийском молодежном научном форуме «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 9 апреля 2024 г., участие);

4. Молодежном научном-образовательном фестивале «Ступени» в рамках III Всероссийского молодежного научного форума «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 12 апреля 2024 г., 2 место);

5. Юбилейном XX Международной научной конференции студентов, аспирантов и молодых ученых «ПРОСПЕКТ СВОВОДНЫЙ - 2024» (г. Лесосибирск, ЛПИ – филиал СФУ, 17 апреля 2024 г., участие).

Разработка мобильного приложения стала ценным опытом, который позволил применить полученные знания и навыки в области разработки мобильных приложений на практике. Созданный проект имеет потенциал для дальнейшего развития и может стать основой для создания коммерчески успешного приложения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Антонов, В. Программирование для Android / В. Антонов. – Москва : Лаборатория знаний, 2019. – 512 с.

2. Бурнет, Э. Привет, Android! Разработка мобильных приложений / Э. Бурнет. – Санкт-Петербург : Питер – 2012. – 253 с.

3. Гайдукова, Е. Базовые знания по IDEF0 / Е. Гайдукова. // Comidware : [сайт] – 2022. – URL: https://clck.ru/34f2Gv (дата обращения: 04.04.2024).

4. Глушенко, С. А., Долженко, А. И. Разработка мобильных приложений: Учебное пособие – Ростов-на-Дону : РГЭУ (РИНХ), 2018. – 221 с.

5. Гриффитс, Д. Head First. Программирование для Android / Д. Гриффитс. – Санкт-Петербург : Питер – 2018. – 912 с.

6. Гришин, А.С. Приложение для вуза Мобильный РГППУ / А.С. Гришин. // Рос. гос. проф.-пед. ун-т, Ин-т инж. - пед. образования, каф. информ. систем и технологий. – Екатеринбург, – 2017. – № 3. – 70 с.

7. Дейтел, П. Android для разработчиков / П. Дейтел. – Санкт-Петербург : Питер, 2016. – 560 с. – 543 с.

8. Евсеев, В. Android. Программирование для профессионалов / В. Евсеев. – Москва : ДМК Пресс, 2017. – 325 с.

9. Елизаров, А. Android. Разработка приложений для чайников / А. Елизаров. – Санкт-Петербург : Питер, 2019. – 246 с.

10. Ермаков, А. Разработка мобильных приложений под Android / А. Ермаков. – Москва : ДМК Пресс, 2019. – 79 с.

11. Ефимов, А. Программирование под Android / А. Ефимов. – Санкт-Петербург : БХВ-Петербург, 2018. – 342 с.

12. Инкрементная модель // Байтэкс : [сайт] – 2017. – URL: https://clck.ru/34cBJ5 (дата обращения: 08.04.2024).

13. Карпов, Ю.Н Android. Программирование для профессионалов / Ю.Н. Карпов. – Санкт-Петербург : Питер, 2019 г. – 297 с.

14. Кинзябулатов, Р В чём различия нотаций IDEF0, DFD и BPMN2.0 / Р. Кинзябулатов // Тринион : [сайт] – 2023. – URL: https://clck.ru/ec648 (дата обращения: 04.04.2024).

15. Колисниченко, Д. Н. Самоучитель. Программирование для Android / Д. Н. Колисниченко. – Москва : БХВ, 2021. – 288 с.

16. Красивская, А. Тестирование мобильных приложений инструкция для начинающих / А. Красивская, А. Коротеева // Блог Яндекс.Практикума : [сайт] – 2022. – URL: https://clck.ru/34f2cM (дата обращения: 25.04.2024).

17. Куликов, С. С. Тестирование программного обеспечения. Базовый курс / С. С. Куликов. – Беларусь : Минск, 2020. – 314с.

18. Ларкин, Г. Программирование на Java для Android. Подробное руководство для разработчика / Г. Ларкин. – Москва : Диалектика, 2018. – 354 с.

19. Левин, Д. Разработка приложений для Android. От простого к сложному / Д. Левин. – Москва : БХВ-Петербург, 2020. – 423 с.

20. Мартин, Р. Чистый код. Создание, анализ и рефакторинг / Р. Мартин. – Санкт-Петербург : Питер, 2019 г. – 298 с.

21. Машнин, Т. С. Разработка Android приложений в деталях / Т. С. Машнин. – Екатеринбург: Издательские решения, 2021. – 400 с.

22. Мищенко, А. В. Бакалаврская работа на тему «Разработка программного обеспечения для клиент-серверного мобильного приложения и web-приложения по истории ТГУ» / А.В. Мищенко. – Тольятти, 2019. – 42 с.

23. Нотация IDEF0 // BisnessStudio : [сайт] – 2019. – URL: https://clck.ru/ec648 (дата обращения: 18.04.2024).

24. Отличие нативных мобильных приложений от всех остальных – // AppMaster : [сайт]. – 2024 – URL: https://appmaster.io/ru/blog/otlichie-nativnyh-mobilnyh-prilozhenij-ot-vseh-ostalnyh (дата обращения: 14.06.2024).

25. Парамонов И. В. Разработка мобильных приложений для платформы Android: учебное пособие / И. В. Парамонов; Яросл. гос. ун-т им. П. Г. Демидова. – Ярославль : ЯрГУ, 2013. – 88 с.

26. Удовенко, М. Android: практическое руководство для разработчиков / М. Удовенко. – Новосибирск : Сибирское университетское издательство, 2018. – 410 с.

27. Ушаков, Е. Основы создания Android-приложений / Е. Ушаков. – Екатеринбург : У-Фактория, 2021. – 360 с.

28. Android Studio: официальный сайт. – 2024. – URL: https://clck.ru/Gjs9Y (дата обращения: 25.04.2024).

29. Appcelerator Titanium – новая RIA платформа // Хабр : [сайт] – 2023. – URL: https://clck.ru/34jSyH (дата обращения: 03.05.2024).

30. Cognex Corporation Приложение «Barcode Scanner» // Google Play : [сайт] – URL: https://clck.ru/34iHmg (дата обращения: 01.05.2024).

31. Clean Architecture in Android and IOS // Apptractor : [сайт] – URL: https://www.s0caler.com/topics/android/clean-architecture-android (дата обращения: 14. 06. 2024).

32. Gamma Play Приложение «Сканер QR и штрих-кодов» // Google Play : [сайт] – URL: https://clck.ru/Em3N7 (дата обращения: 01.05.2024).

33. Pattanayak, P. Google Drive (Google sheet) as Realtime Database as Firebase Android Project Java / P. Pattanayak. // Google Drive : [сайт] – 2020. – URL: https://clck.ru/34f9eH (дата обращения: 01.05.2024).

34. Robbins, A. Разработка информационных систем / A. Robbins. – Москва : Диалектика, 2017. – 608 с.

35. React Native : официальный сайт. – 2024. – URL: https://reactnative.dev/docs/getting-started (дата обращения: 03.05.2024).

36. StartAndroid : [сайт] – 2020. – URL: https://startandroid.ru/ru/ (дата обращения: 03.05.2024).

37. Unity : официальный сайт. – 2018. – URL: https://clck.ru/Tf2J (дата обращения: 03.05.2024).

38. Visual Studio : официальный сайт. – 2020. – URL: https://clck.ru/34jT5S (дата обращения: 03.05.2024).

39. XCode : официальный сайт. – 2023. – URL: https://clck.ru/34jT5w (дата обращения: 03.05.2024).

40. Xamarin : официальный сайт. – 2021. – URL: https://clck.ru/34WKDM (дата обращения: 03.05.2024).

ПРИЛОЖЕНИЕ А
Листинг DatabaseHelper

```
package com.example.park;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHelper extends SQLiteOpenHelper {
    public static final String DATABASE_NAME = "login.db";



    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE user(ID INTEGER PRIMARY KEY
AUTOINCREMENT, username TEXT, password TEXT)");
    }
    public boolean Insert(String username, String password){
        SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put("username", username);
        contentValues.put("password", password);
        long result = sqLiteDatabase.insert("user", null, contentValues);
```

```
        if(result == -1){
            return false;
        }else{
            return true;
        }
    }

    public Boolean CheckUsername(String username){
        SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();
        Cursor cursor = sqLiteDatabase.rawQuery("SELECT * FROM user WHERE
username=?", new String[]{username});
        if(cursor.getCount() > 0){
            return false;
        }else{
            return true;
        }
    }
```

ПРИЛОЖЕНИЕ Б
Листинг MainActivity

```
package com.example.park;

import android.content.Intent;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    DatabaseHelper databaseHelper;

    EditText et_username, et_password, et_cpassword;
    Button btn_register, btn_login;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        databaseHelper = new DatabaseHelper(this);
        et_username = (EditText)findViewById(R.id.et_username);
        et_password = (EditText)findViewById(R.id.et_password);
        et_cpassword = (EditText)findViewById(R.id.et_cpassword);
        btn_register = (Button)findViewById(R.id.btn_register);
        btn_login = (Button)findViewById(R.id.btn_login);
```

```
        btn_login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, Login.class);
                startActivity(intent);
            }
        });

        btn_register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = et_username.getText().toString();
                String password = et_password.getText().toString();
                String confirm_password = et_cpassword.getText().toString();

                if(username.equals("") || password.equals("") ||
confirm_password.equals("")){
                    Toast.makeText(getApplicationContext(), "Fields Required",
Toast.LENGTH_SHORT).show();
                }else{
                    if(password.equals(confirm_password)){
                        Boolean checkusername =
databaseHelper.CheckUsername(username);
                        if(checkusername == true){
                            Boolean insert = databaseHelper.Insert(username, password);
                            if(insert == true){
                                Toast.makeText(getApplicationContext(), "Registered",
Toast.LENGTH_SHORT).show();
```

```
                    et_username.setText("");
                    et_password.setText("");
                    et_cpassword.setText("");
                }
            }else{
                Toast.makeText(getApplicationContext(), "Username already
taken", Toast.LENGTH_SHORT).show();
            }
        }else{
            Toast.makeText(getApplicationContext(), "Password does not match",
Toast.LENGTH_SHORT).show();
        }
    }
}

public class MainActivity extends AppCompatActivity {
    DatabaseHelper databaseHelper;

    EditText et_username, et_password, et_cpassword;
    Button btn_register, btn_login;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        databaseHelper = new DatabaseHelper(this);
        et_username = (EditText)findViewById(R.id.et_username);
        et_password = (EditText)findViewById(R.id.et_password);
```

```
        et_cpassword = (EditText)findViewById(R.id.et_cpassword);
        btn_register = (Button)findViewById(R.id.btn_register);
        btn_login = (Button)findViewById(R.id.btn_login);

        btn_login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, Login.class);
                startActivity(intent);
            }
        });

        btn_register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = et_username.getText().toString();
                String password = et_password.getText().toString();
                String confirm_password = et_cpassword.getText().toString();

                if(username.equals("") || password.equals("") ||
confirm_password.equals("")){
                    Toast.makeText(getApplicationContext(), "Fields Required",
Toast.LENGTH_SHORT).show();
                }else{
                    if(password.equals(confirm_password)){
                        Boolean checkusername =
databaseHelper.CheckUsername(username);
                        if(checkusername == true){
                            Boolean insert = databaseHelper.Insert(username, password);
```

```
                    if(insert == true){
                        Toast.makeText(getApplicationContext(), "Registered",
Toast.LENGTH_SHORT).show();
                        et_username.setText("");
                        et_password.setText("");
                        et_cpassword.setText("");
                    }
                }else{
                    Toast.makeText(getApplicationContext(), "Username already
taken", Toast.LENGTH_SHORT).show();
                }
            }else{
                Toast.makeText(getApplicationContext(), "Password does not match",
Toast.LENGTH_SHORT).show();
            }
            Toast.makeText(getApplicationContext(), "Fields Required",
Toast.LENGTH_SHORT).show();
        }else{
            if(password.equals(confirm_password)){
                Boolean checkusername =
databaseHelper.CheckUsername(username);
                if(checkusername == true){
                    Boolean insert = databaseHelper.Insert(username, password);
                    if(insert == true){
                        Toast.makeText(getApplicationContext(), "Registered",
Toast.LENGTH_SHORT).show();
                        et_username.setText("");
                        et_password.setText("");
                        et_cpassword.setText("");
```

```
                }
            }else{
                Toast.makeText(getApplicationContext(), "Username already
taken", Toast.LENGTH_SHORT).show();
            }
        });
```

ПРИЛОЖЕНИЕ В
Листинг Login

```
package com.example.park;

import android.content.Intent;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class Login extends AppCompatActivity {
    Button btn_lregister, btn_llogin;
    EditText et_lusername, et_lpassword;

    DatabaseHelper databaseHelper;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        databaseHelper = new DatabaseHelper(this);
        et_lusername = (EditText)findViewById(R.id.et_lusername);
        et_lpassword = (EditText)findViewById(R.id.et_lpassword);
        btn_llogin = (Button)findViewById(R.id.btn_llogin);
        btn_lregister = (Button)findViewById(R.id.btn_lregister);
        btn_lregister.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
        public void onClick(View v) {
            Intent intent = new Intent(Login.this, MainActivity.class);
            startActivity(intent);
        }
    });
    btn_llogin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String username = et_lusername.getText().toString();
            String password = et_lpassword.getText().toString();
            Boolean checklogin = databaseHelper.CheckLogin(username, password);
            if(checklogin){
                Toast.makeText(getApplicationContext(), "Login Successful",
Toast.LENGTH_SHORT).show();
            }else{
                Toast.makeText(getApplicationContext(), "Invalid username or
password", Toast.LENGTH_SHORT).show();
            }
            Intent intent = new Intent(Login.this, Prilogenie.class);
            startActivity(intent);
        }
    });
    btn_llogin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String username = et_lusername.getText().toString();
            String password = et_lpassword.getText().toString();
            Boolean checklogin = databaseHelper.CheckLogin(username, password);
            if(checklogin){
```

```
                Toast.makeText(getApplicationContext(), "Login Successful",
Toast.LENGTH_SHORT).show();
            }else{
                Toast.makeText(getApplicationContext(), "Invalid username or
password", Toast.LENGTH_SHORT).show();
            }
            Intent intent = new Intent(Login.this, Prilogenie.class);
            startActivity(intent);
        }
    });
```