|  |  |  |  |
| --- | --- | --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН  СЕВЕРО-КАЗАХСТАНСКИЙ УНИВЕРСИТЕТ ИМ. М. КОЗЫБАЕВА | | | |
|  | | | |
|  | | |  |
| СЕМЕСТРОВЫЙ ПРОЕКТ  На тему: «Репозиторий ретро-игр» | | | |
| Выполнили студенты  группы АПО-20-2 |  | Симонов Д.А.  Сочилович Д.А.  Тулинов А.Е.  Толстов В.Ю. | |
| Доктор PHD, доцент кафедры ИКТ |  | Астапенко Н.В. | |
|  |  |  | |

Петропавловск, 2023

# ЗАДАНИЕ ПО ВЫПОЛНЕНИЮ ПРОЕКТА

Командная разработка программного средства на основе курса Aframe.

В процессе работы команда:

1. Осваивает работу с coda.io

2. Определяет тему проекта исходя из социальной или бизнес актуальности.

3. Планирует процесс разработки, распределяет работы.

3. Выбирает модель и методы разработки из методологий Agile.

4. Разрабатывает UserStory и архитектуру ПС.

5. Реализует проект с помощью систем No-/Low-code / Реализует проект на основе курса Aframe.

СОДЕРЖАНИЕ

[ЗАДАНИЕ ПО ВЫПОЛНЕНИЮ ПРОЕКТА 2](#_Toc133871635)

[ВВЕДЕНИЕ 4](#_Toc133871636)

[1 Аналитическая часть 6](#_Toc133871637)

[1.1 Методология разработки Scrum 6](#_Toc133871638)

[1.2 Технология Coda.io 8](#_Toc133871639)

[1.3 Технология Bravo Studio 10](#_Toc133871640)

[1.4 Распределение обязанностей 13](#_Toc133871641)

[2 Проектная часть 16](#_Toc133871642)

[2.1 Процесс разработки 16](#_Toc133871643)

[2.1.1 Разработка функции «каталог игр» 16](#_Toc133871644)

[2.1.2 Разработка функции «поиск» 17](#_Toc133871645)

[2.1.3 Разработка функции «оплата» 18](#_Toc133871646)

[2.1.4 Разработка функции «добавление в избранное» 18](#_Toc133871647)

[2.1.5 Разработка функции «скачивание» 19](#_Toc133871648)

[2.1.6 Разработка функции «авторизация» 20](#_Toc133871649)

[2.2 Функционал приложения 21](#_Toc133871650)

[ЗАКЛЮЧЕНИЕ 28](#_Toc133871651)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 29](#_Toc133871652)

# ВВЕДЕНИЕ

В настоящее время игры настолько востребованы, что их различные варианты появляются на различных платформах. Но даже с таким множеством вариантов, некоторые классические игры по-прежнему остаются великолепными и популярными. Именно поэтому мы разработали мобильное приложение, которое позволяет насладиться классическими ретро-играми на смартфонах.

Мобильное приложение предоставляет репозиторий ретро-игр, который включает в себя широкий спектр игр, включая известные игры, такие как Pac-Man, Sonic the Hedgehog, Super Mario Brothers и другие. Наше приложение является полезным для людей, которые хотят поиграть в эти игры, но не имеют доступа к старым игровым приставкам или не хотят тратить деньги на приобретение новых.

Актуальность нашего приложения заключается в том, что люди, особенно молодое поколение, по-прежнему находят интерес в классических ретро-играх, и эти игры имеют огромную историческую ценность в мире видеоигр. Поэтому наш проект имеет своей целью сохранение и продвижение этих игр.

Предметом нашего исследования является создание мобильного приложения, которое предоставляет репозиторий ретро-игр, и его дальнейшее развитие для удовлетворения потребностей пользователей в играх.

Объектом нашего исследования является процесс разработки мобильного приложения, включая использование различных инструментов и технологий, таких как Bravo Studio, Airtable, Firebase и Figma.

В целом, наше мобильное приложение предоставляет возможность насладиться классическими ретро-играми на смартфоне, что делает его очень полезным для людей, которые хотят поиграть в эти игры и насладиться их красочной графикой и увлекательным игровым процессом.

Для создания нашего мобильного приложения мы использовали различные инструменты и технологии, такие как Bravo Studio, Airtable, Firebase и Figma. Bravo Studio использовался для разработки макетов и создания интерфейса приложения. Airtable использовался для управления базой данных ретро-игр, включая информацию о названии игры, дате выпуска, жанре и других характеристиках. Firebase использовался для регистрации и аутентификации пользователей, а также для хранения информации об игроках и их достижениях. Figma использовался для разработки дизайна и прототипирования приложения.

Одним из основных преимуществ использования Bravo Studio является возможность создания дизайна и интерфейса приложения без необходимости написания кода. Это позволяет сократить время разработки и сосредоточиться на других аспектах проекта. Кроме того, Bravo Studio интегрируется с другими инструментами, такими как Figma, что облегчает процесс разработки.

Airtable был использован для создания базы данных ретро-игр. Он предоставляет интуитивно понятный интерфейс для создания и управления данными, что облегчает работу с информацией о различных играх.

Firebase использовался для регистрации и аутентификации пользователей, что обеспечивает безопасность приложения и защиту от несанкционированного доступа. Кроме того, Firebase предоставляет удобный способ хранения и управления данными, что облегчает работу с информацией об игроках и их достижениях.

Figma использовался для разработки дизайна и прототипирования приложения. Этот инструмент позволяет создавать профессиональные дизайны интерфейса, которые можно легко прототипировать и исправлять.

В целом, использование этих инструментов и технологий позволило нам создать мобильное приложение, которое предоставляет репозиторий ретро-игр с удобным интерфейсом, безопасностью и хорошо организованной базой данных.

# 1 Аналитическая часть

# 1.1 Методология разработки Scrum

Scrum – это методология гибкой разработки программного обеспечения, которая призвана увеличить производительность команды и качество продукта, используя итеративный и инкрементальный подход к разработке. Основными принципами Scrum являются:

* Работа в итерациях (спринтах): команды работают в коротких, фиксированных по времени итерациях, обычно продолжительностью от 1 до 4 недель, в течение которых они создают инкремент продукта, который готов к использованию.
* Командная работа: команда разработки в Scrum состоит из небольшой группы людей, которые работают вместе для достижения общей цели. Команды самоорганизующиеся, они принимают решения вместе и делятся ответственностью за результат.
* Backlog продукта: это список требований к продукту, который составляет владелец продукта. Он описывает то, что нужно сделать, чтобы продукт был готов к выходу на рынок. Backlog продукта постоянно обновляется, чтобы отразить изменения в требованиях к продукту.
* Backlog спринта: это список задач, которые команда планирует выполнить в течение спринта. Задачи выбираются из Backlog продукта, и они должны быть выполнены командой в течение спринта.
* Daily Scrum: это ежедневное собрание команды, которое проходит на 15 минут и на котором каждый член команды отвечает на три вопроса: что он сделал вчера, что он собирается сделать сегодня и какие препятствия у него возникли.
* Sprint Review: это обзор продукта, который проводится в конце каждого спринта. Команда представляет результат своей работы за спринт, владелец продукта дает обратную связь, и команда обсуждает, что они могут улучшить в следующем спринте.
* Sprint Retrospective: это собрание, которое проводится в конце каждого спринта и на котором команда обсуждает, что они сделали хорошо, что можно улучшить и как они могут улучшить свой процесс разработки в целом.

В Scrum есть также роли: владелец продукта, Scrum-мастер и команда разработки. Владелец продукта ответственен за Backlog продукта и работает с командой, чтобы определить, какие функции продукта должны быть реализованы. Он также управляет приоритетами задач в Backlog продукта и работает с командой, чтобы убедиться, что продукт соответствует требованиям пользователей.

Scrum-мастер отвечает за обеспечение процесса Scrum в команде и устранение препятствий, которые могут повлиять на работу команды. Он также работает с командой над улучшением процесса Scrum, помогает владельцу продукта в создании и управлении Backlog продукта и обучает команду принципам Scrum.

Команда разработки отвечает за создание продукта в течение спринта и достижение целей, установленных на спринт. Команда самостоятельно выбирает задачи, которые они будут выполнять в течение спринта, и работает вместе, чтобы достигнуть целей.

Scrum предоставляет командам гибкую и прозрачную методологию для разработки продукта. Она позволяет команде быстро адаптироваться к изменениям в требованиях пользователя, а также позволяет команде работать более эффективно и продуктивно.

Команда проводит планирование спринта на начало каждого спринта, чтобы определить, какие задачи из Backlog продукта будут выполнены в течение спринта, оценить необходимые ресурсы и установить цели для достижения в конце спринта.

Во время спринта команда проводит короткие ежедневные встречи, называемые Daily Scrum, чтобы обсудить прогресс и препятствия, с которыми столкнулась команда при выполнении задач.

В конце спринта команда проводит обзор спринта, на котором демонстрирует выполненную работу и обсуждает уроки, которые были извлечены из опыта команды.

Кроме того, команда проводит Retrospective во время обзора спринта, чтобы обсудить, как они могут улучшить свой процесс Scrum и повысить свою эффективность в будущем.

После Retrospective команда начинает следующий спринт и процесс повторяется.

Scrum также позволяет команде работать над реализацией сложных задач с помощью разбиения их на более простые и управляемые части. Каждая часть может быть реализована отдельно, а затем объединена в единый функциональный продукт. Такой подход позволяет более гибко управлять процессом разработки.

Одним из основных принципов Scrum является постоянное улучшение процесса. Команда регулярно анализирует свою работу, чтобы найти пути улучшения процесса разработки, реализации задач и общения внутри команды. Это помогает команде постоянно улучшать свою эффективность и достигать лучших результатов.

Применение Scrum также способствует более прозрачной и открытой коммуникации внутри команды. Каждый участник команды знает, что от него ожидается в рамках каждого спринта и какая работа уже выполнена. Это также способствует более прозрачному общению между командой, что позволяет достигать лучших результатов и повышать качество продукта.

Кроме того, Scrum позволяет команде быстро реагировать на изменения в требованиях пользователей. Например, если заказчик внесет изменения в требования в середине спринта, команда сможет легко перепланировать свою работу и продолжить работу над продуктом.

В целом, применение Scrum помогает команде разработки мобильного приложения достигать лучших результатов, улучшать процесс разработки, повышать качество продукта и более эффективно работать в команде.

Мы выбрали Scrum как методологию разработки мобильного приложения, потому что она позволяет нам более эффективно работать в команде и достигать лучших результатов. С помощью Scrum мы можем более гибко управлять процессом разработки. Кроме того, методология Scrum способствует более прозрачной и открытой коммуникации внутри команды, что помогает достигать лучших результатов и повышать качество продукта. Наконец, постоянное улучшение процесса является одним из основных принципов Scrum, что позволяет нам постоянно улучшать свою эффективность и достигать лучших результатов в разработке приложения.

# 1.2 Технология Coda.io

Coda.io – это онлайн-сервис, который объединяет функциональность электронных таблиц, баз данных, документов и других инструментов в единый рабочий пространство. Он позволяет создавать документы, проекты и таблицы с уникальными функциями, интегрировать их с другими приложениями и сервисами, а также работать над ними совместно с командой в режиме реального времени.

Основные преимущества Coda.io:

* Гибкость и универсальность: Coda.io позволяет создавать документы любой сложности, включая таблицы, базы данных, справочники и другие типы документов. Пользователь может создавать любые функции, которые вам нужны, используя формулы, а также расширять функциональность Coda.io с помощью сторонних интеграций.
* Коллаборация в режиме реального времени: Coda.io позволяет работать с документами в режиме реального времени с другими участниками команды. Пользователь может видеть изменения, вносимые другими участниками, и обсуждать их встроенными инструментами комментирования и обсуждения.
* Удобство использования: Coda.io предоставляет простой и интуитивно понятный пользовательский интерфейс, что делает его доступным для использования для любого уровня навыков пользователей. Он также имеет множество готовых шаблонов, которые могут быть использованы для быстрого создания документов и таблиц.
* Интеграции: Coda.io может интегрироваться с множеством сторонних приложений и сервисов, включая Google Sheets, Slack, Zapier, GitHub и многие другие. Это позволяет автоматизировать множество задач и создавать более эффективные рабочие процессы.
* Сохранение истории изменений: Coda.io автоматически сохраняет историю изменений, что позволяет пользователям отслеживать, кто и когда внес изменения в документ или таблицу.
* Безопасность: Coda.io предоставляет высокий уровень безопасности для хранения и обработки данных. Он использует шифрование данных в покое и в передаче, а также предоставляет возможность управлять уровнем доступа к документам и таблицам.

В целом, Coda.io – это мощный инструмент для создания и управления документами и таблицами, который позволяет пользователям создавать настраиваемые рабочие процессы и работать с командой в режиме реального времени.

Мы используем Coda.io для управления задачами, сроками и коммуникацией внутри команды. Вот как мы смогли использовать Coda.io для этого:

* Создание таблицы задач: Мы можем создать таблицу в Coda.io, которая содержит все задачи, необходимые для завершения проекта, с указанием приоритетов и дат завершения. Мы можем использовать такие колонки, как "Задача", "Описание", "Ответственный", "Срок", "Статус" и другие.
* Назначение задач: Когда мы начинаем работу над проектом, мы можем назначать задачи для каждого члена команды, используя соответствующую колонку в таблице задач. Каждый член команды может видеть, какие задачи ему были назначены, и установить себе напоминания о сроках выполнения.
* Слежение за прогрессом: Мы можем использовать столбец "Статус" в таблице задач, чтобы отслеживать прогресс выполнения задачи. Это позволяет всей команде видеть, насколько близко мы к завершению проекта, а также выявлять возможные задержки.
* Создание таблицы коммуникации: Кроме того, мы можем создать таблицу для коммуникации, где мы будем обмениваться идеями, задавать вопросы и обсуждать проблемы, связанные с проектом. Это позволит нам организовать общение внутри команды, сохранить историю переписки и быстро находить нужную информацию.
* Использование шаблонов: Мы можем использовать готовые шаблоны, доступные в Coda.io, для создания документов, которые могут использоваться в нашем проекте. Например, мы можем использовать шаблон для создания графика Ганта, чтобы отслеживать сроки выполнения задач.

Мы выбрали Coda.io, потому что это мощный инструмент, который предоставляет широкие возможности для управления проектами, коммуникации внутри команды и интеграции с другими приложениями. Все необходимые функции доступны в одном месте, что упрощает и ускоряет работу над проектом.

Кроме того, Coda.io предоставляет готовые шаблоны и инструменты для создания собственных шаблонов, что позволило нам быстро создавать документы, таблицы и другие элементы проекта, не тратя много времени на настройку.

Также стоит отметить, что Coda.io обладает высокой гибкостью, что позволило нам настроить инструмент под наши конкретные потребности и варианты использования. Это означает, что мы смогли адаптировать его к нашей командной работе и проекту.

Кроме того, Coda.io постоянно обновляется и улучшается, добавляя новые функции и возможности. Это позволяет нам быть на шаг впереди и использовать новые функции, чтобы оптимизировать нашу командную работу.

В целом, мы выбрали Coda.io, потому что это надежный и мощный инструмент, который помогает нам эффективно управлять проектом, организовать коммуникацию внутри команды и интегрироваться с другими приложениями, что в свою очередь позволяет нам достичь поставленных целей быстрее и эффективнее.

# 1.3 Технология Bravo Studio

Bravo Studio – это платформа для создания мобильных приложений без необходимости знания кода. Она позволяет людям создавать и опубликовывать приложения на iOS и Android устройствах, используя простой и интуитивно понятный интерфейс.

Платформа имеет редактор, который позволяет пользователям создавать и настраивать макеты приложений, добавлять различные элементы дизайна, такие как кнопки, текст, изображения, видео и многое другое. Пользователи могут также настраивать функциональность приложения, добавлять различные модули, такие как формы, списки, карты и другие, а также создавать и настраивать базы данных для хранения информации.

Bravo Studio поддерживает интеграцию с различными сервисами, такими как Google Maps, Instagram, Twitter, Facebook и другие. Это позволяет пользователям добавлять функциональность, связанную с этими сервисами, в свои приложения.

После того, как приложение создано, пользователи могут опубликовать его в App Store и Google Play Store, где оно становится доступным для скачивания и использования.

Bravo Studio предоставляет различные тарифные планы, в зависимости от потребностей пользователей. Они варьируются от бесплатного плана, который позволяет создавать одно приложение, до более дорогих планов, которые предоставляют более широкие возможности и дополнительную поддержку.

Кроме того, Bravo Studio предоставляет ресурсы и учебные материалы для помощи пользователям в создании своих приложений. Это включает в себя документацию, видеоуроки и сообщество пользователей, где можно задавать вопросы и общаться с другими разработчиками.

Bravo Studio также позволяет пользователям работать в режиме реального времени, что означает, что изменения в приложении могут быть видны непосредственно в редакторе. Это позволяет быстро тестировать различные версии приложений и делать корректировки до публикации в магазинах приложений.

В первую очередь мы можем использовать Bravo Studio для создания макета приложения. Мы можем выбрать из множества готовых элементов дизайна, таких как кнопки, поля для ввода текста, изображения и другие, и настроить их в соответствии с нашими потребностями.

В процессе работы мы можем использовать функцию редактирования в режиме реального времени, чтобы быстро тестировать различные версии приложения и делать корректировки.

Кроме того, мы можем использовать аналитику и пользовательские данные, предоставляемые Bravo Studio, чтобы отслеживать успех нашего приложения и определять области для улучшения.

Мы выбрали Bravo Studio в разработке нашего проекта по нескольким причинам.

Во-первых, Bravo Studio предоставляет интуитивно понятный интерфейс, который позволяет людям без технических знаний и опыта в программировании создавать мобильные приложения. Мы можем использовать готовые элементы дизайна и настраивать их в соответствии с нашими потребностями, а также добавлять функциональность без необходимости вручную писать код.

Во-вторых, Bravo Studio позволяет работать в режиме реального времени, что позволяет быстро тестировать и корректировать приложение.

В-третьих, Bravo Studio предоставляет пользовательскую аналитику и данные, которые помогают отслеживать успех нашего приложения и определять области для улучшения.

В целом, мы выбрали Bravo Studio, потому что она предоставляет мощный инструмент для создания мобильных приложений, который позволяет нам создавать высококачественные приложения без необходимости вручную писать код.

Для создания макетов, а также оформления дизайна использовалась среда Figma. Это инструмент для создания макетов и прототипов веб-приложений и мобильных приложений, который позволяет дизайнерам и разработчикам работать в режиме реального времени над одним и тем же проектом. Figma имеет широкий набор функций, включая возможность создания векторных изображений, добавления элементов интерфейса и создания интерактивных прототипов.

Мы использовали Figma для создания макетов и прототипов нашего приложения вместе с Bravo Studio. С помощью Figma мы могли создавать дизайн интерфейса нашего приложения, добавлять элементы и определять стиль, цвет и шрифты. Мы также использовали Figma для создания прототипов нашего приложения, чтобы проверить работоспособность и дизайн приложения перед его разработкой в Bravo Studio.

С помощью Figma мы могли легко создавать макеты и прототипы приложения в режиме реального времени, что позволяло всей команде сотрудничать и работать вместе над проектом. Мы использовали возможность Figma для создания отдельных страниц и проектов, чтобы разделять дизайн на разные компоненты, такие как логин, профиль пользователя и главная страница приложения.

После того, как мы создали макеты и прототипы приложения в Figma, мы экспортировали их в форматы, которые совместимы с Bravo Studio. Это позволило нам импортировать дизайн в Bravo Studio и использовать его в разработке приложения.

В качестве базы данных в разработке проекта используется Airtable API. Airtable – это облачная база данных, которая позволяет создавать, хранить и управлять данными любой сложности. Она предоставляет пользовательский интерфейс для создания таблиц, которые можно использовать для хранения данных, таких как контакты, задачи, проекты, продукты и многое другое. Кроме того, Airtable позволяет настраивать поля таблицы и взаимодействовать с другими инструментами, такими как Bravo Studio. Bravo Studio позволяет легко интегрировать данные из Airtable в мобильные приложения, созданные в Bravo Studio.

При совместном использовании Airtable и Bravo Studio, мы можем управлять данными и быстро обновлять данные в приложении, при этом не нужно вручную писать код для интеграции данных. Это значительно упрощает и ускоряет процесс разработки, так как мы можем сосредоточиться на разработке функциональности приложения, а не на написании кода для работы с данными.

Кроме того, Airtable предоставляет различные инструменты для управления данными, такие как фильтры, сортировки и формулы, которые мы можем использовать для настройки отображения и манипулирования данными в нашем приложении.

В целом, Airtable и Bravo Studio предоставляют нам мощный инструментарий для создания мобильных приложений, которые быстро и эффективно работают с данными. Мы можем использовать Airtable для управления и хранения данных, а затем интегрировать эти данные в наше мобильное приложение, созданное в Bravo Studio, чтобы получить мощное приложение, которое быстро обновляется с новыми данными и удовлетворяет требованиям пользователей.

Для добавления в приложения функциональности входа и аутентификации пользователя мы использовали Firebase. Firebase – это платформа для разработки мобильных и веб-приложений, которая предоставляет широкий набор инструментов для создания приложений, включая инструменты для хранения, аналитики, облачных функций и многое другое. Firebase также предоставляет набор инструментов для регистрации и аутентификации пользователей, которые мы можем использовать для создания безопасных и защищенных приложений.

С помощью Firebase мы можем добавить функционал регистрации и аутентификации в мобильные приложения, созданные в Bravo Studio. Firebase предоставляет API для работы с регистрацией и аутентификацией, которые мы можем использовать в нашем приложении. Мы можем настроить Firebase для подключения к нашему приложению в Bravo Studio, используя соответствующий ключ API. В целом, Firebase был незаменимым инструментом для разработки нашего проекта в Bravo Studio.

# 1.4 Распределение обязанностей

Распределение ответственности в команде разработки проекта играет важную роль в достижении успеха. Оно позволяет определить роли и обязанности каждого члена команды, установить ясные цели и ожидания, а также обеспечить более эффективную и продуктивную работу.

Распределение ответственности позволяет четко определить кто отвечает за конкретные аспекты проекта, что позволяет лучше планировать и контролировать процесс работы. Это также помогает избежать дублирования работ, что может снижать эффективность команды и затягивать сроки.

Кроме того, распределение ответственности способствует улучшению коммуникации в команде. Каждый член команды понимает, кто за что отвечает, и может обратиться к соответствующему человеку при возникновении вопросов. Это также позволяет установить более четкие и реалистичные сроки выполнения задач, что способствует более точному планированию и сокращению задержек. Распределение ответственностей помогает оптимизировать процесс работы, повышает качество и эффективность проекта, а также улучшает коммуникацию и сотрудничество в команде.

Команда разработки приложения состоит из четырех разработчиков. Согласно методологии Scrum, она предполагает следующие роли и их ответственности:

1. Scrum Master:

* Организация ежедневных стендапов;
* Управление бэклогом продукта и планирование спринтов;
* Координация работы команды и поддержка коммуникации между разработчиками;
* Решение конфликтов и проблем внутри команды;

1. Product Owner:

* Определение целей и задач приложения;
* Работа с пользователями для определения требований и пожеланий;
* Составление бэклога продукта и планирование релизов;
* Приоритизация задач и определение приоритетов разработки;

1. Frontend Developer:

* Разработка пользовательского интерфейса;
* Работа с дизайнером для определения функционала и дизайна приложения;
* Настройка блоков и тестирование функционала;

1. Backend Developer:

* Разработка серверной части приложения
* Работа с базами данных и API
* Настройка блоков и тестирование функционала;
* Обеспечение безопасности и надежности работы сервера;

Матрица ответственности согласно методологии Scrum изображена на таблице 1.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Обязанности/Разработчики | Scrum Master | Product Owner | Frontend Developer | Backend Developer |
| Организация стендапов | + |  |  |  |
| Управление бэклогом продукта | + |  |  |  |
| Координация работы команды | + |  |  |  |
| Решение конфликтов | + |  |  |  |
| Определение целей и задач |  | + |  |  |
| Разработка интерфейса |  |  | + |  |
| Настройка блоков |  |  | + | + |
| Тестирование функционала |  |  | + | + |
| Разработка серверной части |  |  |  | + |
| Работа с базами данных |  |  |  | + |
| Обеспечение безопасности |  |  |  | + |

Таблица 1.1 Матрица ответственности

Как видно из матрицы ответственности, каждый член команды имеет свои задачи и обязанности, необходимые для успешного завершения проекта. Все роли в команде имеют равную значимость и взаимодействуют друг с другом для создания приложения высокого качества.

# 2 Проектная часть

# 2.1 Процесс разработки

Приложение «Репозиторий ретро-игр» включает в себе следующие функциональные блоки:

* Каталог игр;
* Поиск
* Добавление в избранное
* Оплата
* Скачивание
* Авторизация

Далее будут подробно рассмотрены этапы разработки для каждого из этих блоков.

# 2.1.1 Разработка функции «каталог игр»

Процесс разработки функционального блока «каталог игр» состоит из следующих этапов:

1. Планирование:

Команда разработчиков обговаривает требования и определяет функциональные возможности, которые должны быть реализованы в каталоге игр. Затем команда обсуждает возможные способы реализации функциональных возможностей и определяет необходимые ресурсы;

1. Проектирование:

На основе обсуждений команда разработчиков создает макеты экранов для каталога игр в Figma. Дизайнер разрабатывает дизайн экранов на основе макетов, и в результате получается прототип, который команда использует в дальнейшей разработке;

1. Настройка Airtable:

Создается база данных игр в Airtable, в которую затем будут добавляться игры для отображения в каталоге. Определяются необходимые поля для каждой игры, такие как название, жанр, год выпуска, описание, обложка и т.д;

1. Разработка бэкенда:

С помощью Firebase команда разработчиков настраивает регистрацию и аутентификацию пользователей. Затем создается серверная логика, которая позволяет получать данные из базы данных Airtable и отображать их в каталоге игр. В частности, команда разработчиков настраивает функцию, которая отображает все игры, добавленные в базу данных Airtable;

1. Разработка фронтенда:

С помощью Bravo Studio команда разработчиков создает экраны для отображения игр в каталоге. Для каждой игры создается отдельный экран, на котором отображается информация о ней, такая как название, обложка, жанр, год выпуска, описание и т.д. Также создается экран поиска игр, где пользователи могут найти игры по жанру, году выпуска и другим параметрам;

1. Тестирование:

После завершения разработки команда разработчиков проводит тестирование каталога игр, чтобы убедиться, что все функциональные возможности работают корректно и что пользовательский интерфейс является интуитивно понятным и удобным в использовании;

# 2.1.2 Разработка функции «поиск»

Разработка функционального блока "поиск" для нашего приложения на Bravo Studio может быть разделена на несколько этапов:

1. Планирование и проектирование функционала поиска:

* Определение параметров поиска, которые должны быть доступны для пользователя (например, название игры, жанр, год выпуска, разработчик);
* Определение логики поиска и фильтрации результатов;
* Проектирование интерфейса для ввода параметров поиска и отображения результатов;

1. Реализация функционала поиска:

* Создание формы для ввода параметров поиска;
* Настройка подключения к базе данных Airtable, чтобы получить доступ к таблице с информацией об играх;
* Написание кода для выполнения запроса к базе данных с учетом выбранных пользователем параметров поиска;
* Обработка результатов запроса и отображение их на странице в удобном формате для пользователя;

1. Тестирование и отладка:

* Проведение тестового поиска с различными параметрами и проверка корректности результатов;
* Исправление ошибок и улучшение производительности.

1. Документация:

* Создание документации для пользователей, которая объясняет, как использовать функционал поиска и какие параметры доступны;

# 2.1.3 Разработка функции «оплата»

Для разработки и реализации платёжной системы были проделаны следующие шаги:

1. Определение требований: Сначала мы определяем, какие требования мы должны удовлетворить, чтобы сделать блок оплаты полезным для пользователей. Например, мы должны предоставить безопасный и удобный способ оплаты, позволяющий пользователям покупать игры в нашем каталоге;
2. Дизайн: Следующим шагом является создание дизайна блока оплаты. Мы можем использовать Figma для создания макетов и прототипов, чтобы определить, как будет выглядеть и работать наш блок оплаты;
3. Разработка на Bravo Studio: После создания макета мы переходим к разработке блока оплаты на платформе Bravo Studio. Мы можем использовать инструменты Bravo Studio для создания пользовательского интерфейса, интеграции с Firebase для обработки оплаты и обработки заказов.
4. Интеграция с Firebase: Для обработки оплаты мы можем использовать Firebase, который предоставляет множество инструментов для обработки платежей. Мы можем использовать Firebase Functions для обработки платежей на стороне сервера, Firebase Authentication для безопасной авторизации пользователей, Firebase Realtime Database для хранения информации о заказах и оплате, а также Firebase Hosting для размещения нашего приложения;
5. Тестирование: После разработки и интеграции блока оплаты мы должны протестировать его, чтобы убедиться, что он работает должным образом и не вызывает проблем у пользователей. Мы можем использовать инструменты тестирования Bravo Studio и Firebase для тестирования блока оплаты.

# 2.1.4 Разработка функции «добавление в избранное»

В процессе создания функции «добавление в избранное» были проделаны следующие этапы:

1. Проектирование интерфейса

Первым этапом является проектирование интерфейса. Для этого мы используем Figma. В этом блоке мы должны создать кнопку "Добавить в избранное", которая будет размещена на странице деталей игры. Также мы должны создать страницу "Избранное", на которой будут отображаться все игры, добавленные пользователем в избранное. На этом этапе также определяются требования к функциональности;

1. Создание базы данных

Для сохранения информации о том, какие игры были добавлены в избранное, мы используем Airtable. На этом этапе мы создаем таблицу, в которой будут храниться данные об играх, добавленных в избранное. Также мы создаем связь между этой таблицей и таблицей с играми;

1. Создание компонента

На этом этапе мы используем Bravo Studio для создания компонента, который будет реализовывать функциональность добавления игр в избранное. Мы создаем действие "Добавить в избранное" и связываем его с кнопкой "Добавить в избранное". При нажатии на эту кнопку, данные об игре сохраняются в таблицу в Airtable;

1. Реализация функциональности "Избранное"

На этом этапе мы создаем страницу "Избранное" и подключаем ее к таблице с играми в Airtable. Мы также создаем функцию, которая будет выводить список игр, добавленных в избранное;

1. Тестирование

Последний этап - тестирование функциональности "Добавление в избранное". Мы проверяем, что кнопка работает корректно и что данные о добавленных играх сохраняются в таблице в Airtable. Мы также проверяем, что страница "Избранное" отображает правильный список игр.

# 2.1.5 Разработка функции «скачивание»

Разработка функции «скачивание» состоит из следующих этапов:

1. Определение требований: необходимо определить требования к функциональности блока "скачивание игр". Например, какие типы игр могут быть загружены, какие форматы файлов поддерживаются, какие механизмы скачивания будут использоваться и т.д.;
2. Проектирование интерфейса: на основе определенных требований необходимо разработать интерфейс блока "скачивание игр". В Figma можно создать макет экрана с кнопкой загрузки и информацией о загруженных файлах;
3. Создание базы данных: в Airtable необходимо создать базу данных для хранения информации о загруженных файлах и связать ее с приложением на Bravo Studio;
4. Создание блока: необходимо создать блок, который будет обрабатывать запросы пользователя на загрузку файлов и сохранение информации о загруженных файлах в базе данных. Для этого можно использовать Firebase Cloud Functions, которые будут запускаться при запросе на загрузку;
5. Тестирование: после написания кода необходимо протестировать его, чтобы убедиться в правильности работы блока "скачивание игр";
6. Внесение изменений: если в процессе тестирования были выявлены ошибки, необходимо внести изменения в код и повторно протестировать блок "скачивание игр";
7. Развертывание: после тестирования и внесения изменений блок "скачивание игр" можно развернуть на сервере и включить его в работу приложения.

# 2.1.6 Разработка функции «авторизация»

В разработку функции авторизации входят следующие этапы:

1. Анализ требований: перед началом разработки необходимо определить, какие данные нужны для авторизации пользователя и как они будут храниться. Также необходимо определить, какие методы авторизации будут доступны (например, через социальные сети, электронную почту или номер телефона);
2. Создание макета интерфейса: после того, как определены требования, необходимо создать макет интерфейса для авторизации. На этом этапе определяются элементы интерфейса, такие как поля для ввода логина и пароля, кнопки и т.д.;
3. Разработка бизнес-логики: на этом этапе определяется, как будет проходить процесс авторизации. Например, при входе в приложение нужно проверять логин и пароль пользователя, а при использовании социальных сетей - получать данные пользователя из API социальной сети;
4. Разработка клиентской части: после разработки бизнес-логики и создания макета интерфейса необходимо начать разработку клиентской части. На этом этапе создаются все необходимые элементы интерфейса, а также взаимодействие с бизнес-логикой;
5. Разработка серверной части: для хранения информации о пользователях, их авторизации и т.д. необходимо создать серверную часть. На этом этапе определяется, какие технологии будут использоваться для хранения данных (например, база данных) и создания серверной части;
6. Тестирование и отладка: после того, как разработка функции авторизации завершена, необходимо провести тестирование и отладку приложения. На этом этапе проверяется, как функция авторизации работает на разных устройствах, в разных ситуациях и т.д.;
7. Внедрение: после успешного тестирования и отладки функция авторизации внедряется в приложение.

Таким образом, в процессе разработки были созданы функциональные блоки, такие как "каталог игр", "поиск", "оплата", "добавление в избранное", "скачивание игр" и авторизация, каждый из которых был тщательно проработан и проверен на работоспособность. В ходе разработки нашего приложения мы использовали инструменты Bravo Studio, Airtable, Firebase и Figma, что значительно ускорило процесс и повысило его эффективность.

# 2.2 Функционал приложения

Страница каталога имеет в себе кнопки переключения вкладок, кнопку «избранное», а также кнопку на переход к своему профилю.

Каталог включает в себя две вкладки – «игры» и «консоли».

Каталог игр изображен на рисунке 2.1



Рисунок 2.1 Каталог игр

Если нажать на кнопку справа от игры, то она добавится в избранное.

Каталог для консолей изображен на рисунке 2.2



Рисунок 2.2 Каталог консолей

Функционал каталога консолей идентичен функционалу каталога игр.

При выборе игры или консоли в каталоге пользователь переходит на страницу соответствующего товара. Страница игры представлена на рисунке 2.3



Рисунок 2.3 Страница игры

Страница игры предоставляет общую информацию об игре, такую как количество игроков, компанию-разработчика, дату выхода, жанр, описание игры, а также оценки критиков и то, на каких платформах игра выходила. В верхней части страницы находятся промоматериалы, скриншоты, трейлеры или кадры из игры. Под этой частью находится кнопка для покупки картриджа, а в нижней части страницы находится кнопка для скачивания ROM-файла игры.

На рисунке 2.4 представлена страница о консоли.



Рисунок 2.4 Страница о консоли

На странице о консоли представлено изображение консоли, страна-производитель, а также сводка информации о самой консоли и совместимой с ней техникой, как например мониторы и кабели. Также в описании сказано, с чем в комплекте приставка поставляется (например, с какими контроллерами).

Следующая функция – оплата.

Оплата и оформление заказа представлены на рисунке 2.5

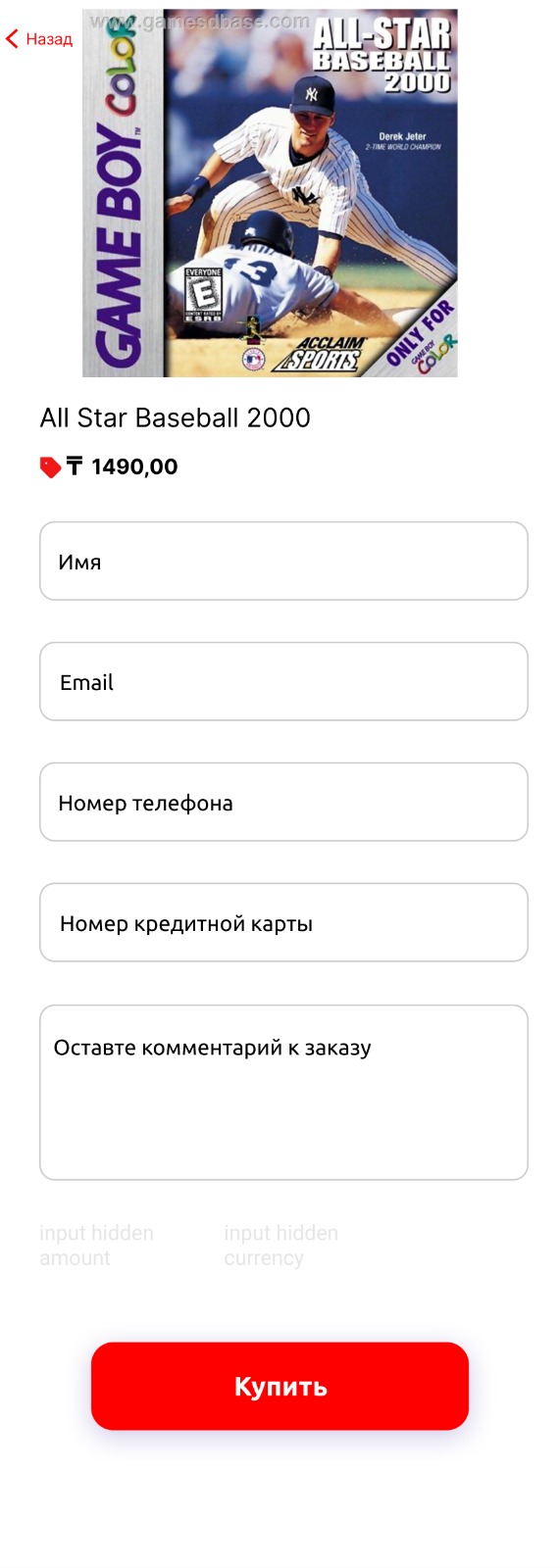


Рисунок 2.5 Меню оформления заказа

Исходя из рисунка, при оформлении заказа пользователь может оставить свои контактные данные, свой номер кредитной карты, а также дополнительно предполагается возможность оставить комментарий к заказу.

Следующая функция – добавление игры в избранное и просмотр списка избранных игр. Меню «избранное» приведено на рисунке 2.6



Рисунок 2.6 Меню «Избранное»

При добавлении игры в избранное, она отображается здесь. Игру можно удалить из избранного в любой момент при повторном нажатии на соответствующую кнопку.

Следующая функция – регистрация и авторизация. Экран регистрации представлен на рисунке 2.7.

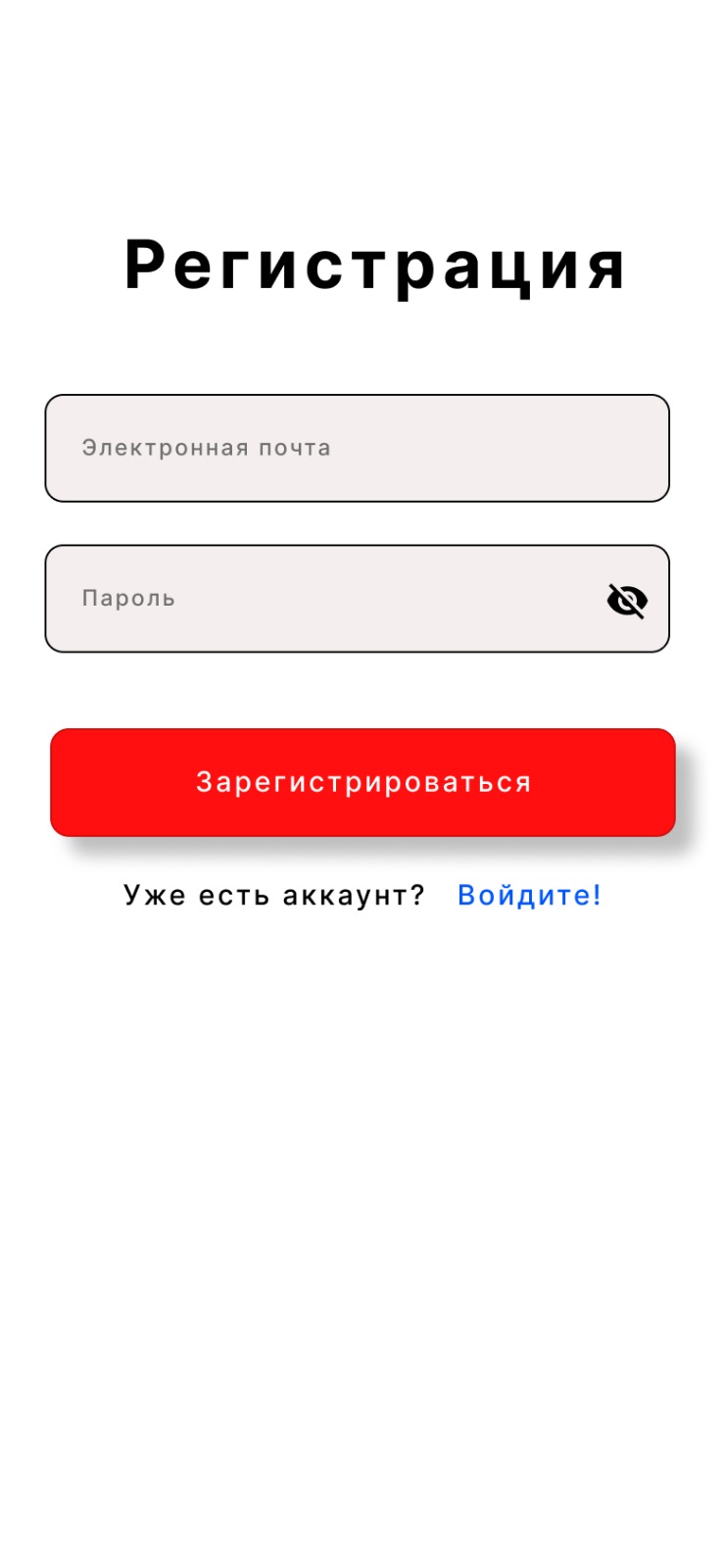


Рисунок 2.7 Регистрация

На экране регистрации пользователь может ввести свою электронную почту и пароль. Также, если у пользователя уже есть аккаунт, под кнопкой «Зарегистрироваться» есть кнопка «Войдите!». Соответственно, есть меню для входа в свой аккаунт, которое представлено на рисунке 2.8

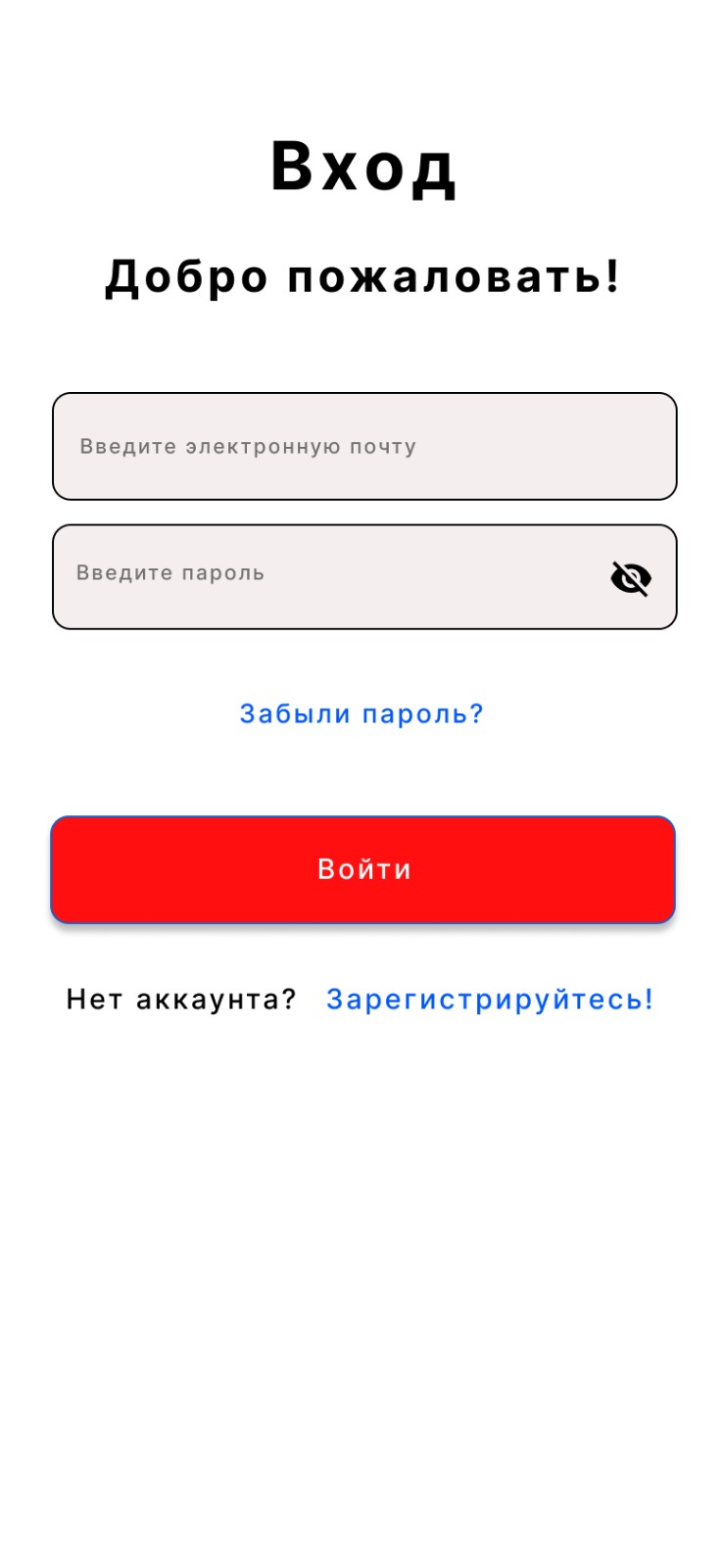


Рисунок 2.9 Меню входа.

На меню входа в аккаунт предполагается ввод своих данных – электронной почты и пароля. Помимо этого, если пользователь не помнит пароль, то есть кнопка восстановления. Также в нижней части этого меню есть перенаправление на форму регистрации.

Если пользователь попытается войти в аккаунт, когда он уже находится в системе, то будет отображено следующее оповещение, которое изображено на рисунке 2.10.

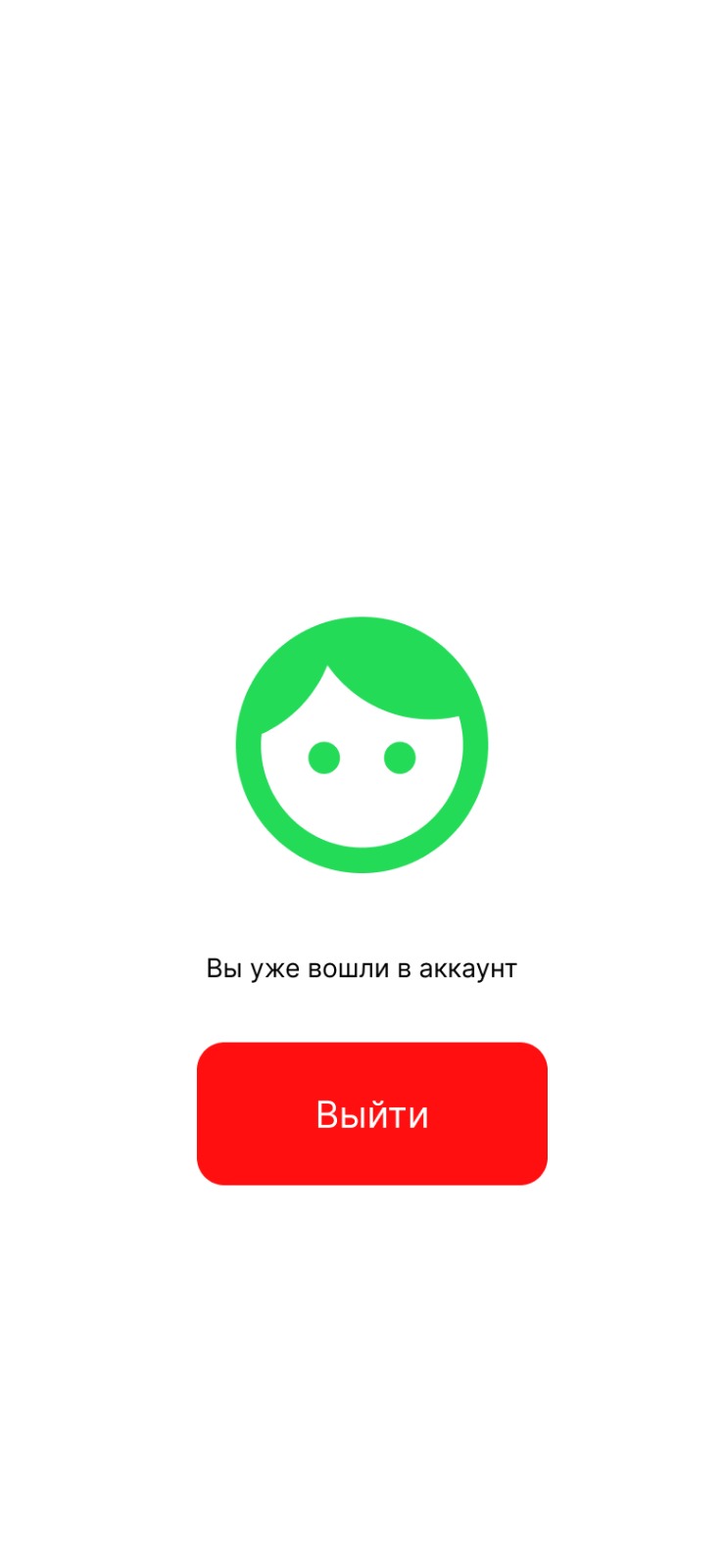


Рисунок 2.10 Оповещение о том, что пользователь в системе

# ЗАКЛЮЧЕНИЕ

В заключение можно сказать, что проект по разработке мобильного приложения "Репозиторий ретро-игр" был успешно завершен. Мы достигли наших целей и задач, которые включали создание приложения, предоставляющего удобный доступ к ретро-играм, с безопасной регистрацией и аутентификацией пользователей, а также с удобной базой данных игр и информации об игроках.

В процессе разработки мы приобрели множество новых навыков и знаний. Мы изучили использование различных инструментов и технологий, таких как Bravo Studio, Airtable, Firebase и Figma. Мы научились создавать дизайн интерфейса, управлять базой данных и обеспечивать безопасность приложения.

Итоговое приложение включает в себя такой функционал как каталог, поиск, оплата, добавление в избранное и авторизация, что позволяет пользователям наслаждаться ретро-играми.

Также стоит отметить, что проект по разработке мобильного приложения "Репозиторий ретро-игр" является актуальным и полезным в наше время, когда многие люди любят играть в ретро-игры, но не всегда могут найти удобный и безопасный способ доступа к ним. Наше приложение предоставляет такой способ, и мы уверены, что оно будет успешным на рынке мобильных приложений.

В целом, проект по разработке мобильного приложения "Репозиторий ретро-игр" был полезным опытом для всей команды. Мы научились работать вместе, решать проблемы и достигать поставленных целей. Мы гордимся тем, что смогли создать такое приложение и уверены, что это только начало нашего пути в мире разработки мобильных приложений.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Ушакова, Е.В. Методические указания по выполнению дипломных проектов/работ для бакалавриата направления информационно-коммуникационные технологии: учебное пособие / Е.В. Ушакова, Л.В. Долматова. – Петропавловск, 2019. -99 с.;

2 Страница обучения Bravo Studio [Электронный ресурс]. – Режим доступа: URL: https://projects.bravostudio.app/tutorials;

3 Официальный сайт Figma [Электронный ресурс]. – Режим доступа: URL: https://www.figma.com/;

4 Начало работы с Airtable [Электронный ресурс]. – Режим доступа: URL: https://support.airtable.com/docs/getting-started-with-airtables-web-api;

5 Документация Firebase [Электронный ресурс]. – Режим доступа: URL: https://firebase.google.com/docs?hl=ru;

6 Документация Coda.io [Электронный ресурс]. – Режим доступа: URL: https://coda.io/resources/learn;

7 Пирская Л.В.: Разработка мобильных приложений в среде Android Studio. – Южный Федеральный Университет, 2019;

8 Танг Дж.: Умные мобильные проекты с TensorFlow. – ДМК Пресс, 2018;

9 Нефедов Ю., Чурсин А., Мамедова Н.: Разработка кроссплатформенных мобильных приложений – перспективные методы и стандартные практики. – Синергия, 2021;

10 Федотенко М.: Разработка мобильных приложений. Первые шаги. – Лаборатория знаний, 2019;

11 Еранссон А.: Эффективное использование потоков в операционной системе Android. – ДМК-Пресс, 2014;

12 Семенчук В.: Мобильное приложение как инструмент бизнеса. – Альпина Диджитал, 2016;

13 Колисниченко Д.: Программирование для Android. 3-е издание. – БХВ-Петербург, 2020;

14 Клифтон Я.: Проектирование пользовательского интерфейса Android. 2-е издание. – ДМК-Пресс, 2017;

15 Нормативные документы, отражающие направления деятельности СКУ им. М.Козыбаева, образовательные программы, реализуемые высшим учебным заведением, обучаемой специальностью, видами, функциями и задачами будущей профессиональной деятельности.