

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»



Направление подготовки
09.03.01 Информатика и вычислительная техника

Направленность (профиль)
«Технологии разработки программного обеспечения»

Выпускная квалификационная работа

Разработка мобильного приложения «Gym Tracker»

Обучающегося 4 курса
очной формы обучения
Игнатьева Дениса Сергеевича

Руководитель выпускной квалификационной
работы:
кандидат педагогических наук, доцент,
доцент кафедры информационных технологий и
электронного обучения
Гончарова Светлана Викторовна

Санкт-Петербург
2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
ГЛАВА 1. ТЕОРЕТИЧЕСКИЙ ПОДХОД К РАЗРАБОТКЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ	5
1.1 Обзор существующих приложений и технологий	5
1.2 Требования к мобильному приложению	15
ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА МОБИЛЬНОГО УПРАЖНЕНИЯ	17
2.1 Прототип пользовательского интерфейса	17
2.2 Создание иконки	19
2.3 Описание процесса разработки	24
2.4 Тестирование	33
2.5 Перспективы	35
ЗАКЛЮЧЕНИЕ	37
СПИСОК ЛИТЕРАТУРЫ	38

ВВЕДЕНИЕ

Хорошая физическая подготовка всегда являлась неотъемлемой частью жизни человека. От доисторических охотников до современных спортсменов. В своем стремлении к физическому совершенству люди применяли различные методы, которые постоянно эволюционировали. В современном мире тренажерные залы стали местом встречи людей разных возрастов и профессий, каждый из которых преследует свои уникальные цели: от формирования идеальной фигуры до развития силы и выносливости, а также подготовки к соревнованиям.

Достижение этих целей становится системой, требующей от человека системный подход и дисциплины, чтобы достичь желаемых результатов. Ведение дневника тренировок существенно облегчает этот процесс, позволяя систематизировать свои усилия. До недавнего времени люди записывали тренировки в блокноты и тетради, чтобы отслеживать прогресс и точно знать, как проходили их предыдущие тренировки.

Однако, с развитием технологий и распространением мобильных устройств подход к тренировочному процессу начал меняться. Вместо того чтобы носить с собой тетрадь или блокнот, пользователи могут вести учёт своих тренировок и прогресса через мобильные приложения, которые часто предлагают дополнительные функции, такие как аналитика, статистика и даже персонализированные рекомендации.

Актуальность данной темы заключается в разработке мобильного приложения, которое поспособствует переходу на мобильные системы учета тренировок.

Предметом исследования выступает мобильное приложение «Gym Tracker».

Практическая значимость заключается в полноценной разработки мобильного приложения, в умении анализировать существующие продукты и выбирать технологии реализации.

Целью бакалаврской выпускной квалификационной работы является разработка мобильного приложения «Gym Tracker».

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих приложений для учёта тренировок;
2. Провести анализ инструментов для реализации приложения;
3. Сформировать требования для приложения;
4. Создать прототип мобильного приложения;
5. Описать процесс создания и тестирования.

Результатом работы является работающее приложение с возможностью записи тренировок.

Выпускная квалификационная работа состоит из теоретической части, где проводится анализ существующих приложений и инструментов для разработки, а также определяется список требований к приложению, и практической, где описывается процесс разработки мобильного приложения. **Всего содержится n страниц, y изображений и x таблицы.**

ГЛАВА 1. ТЕОРЕТИЧЕСКИЙ ПОДХОД К РАЗРАБОТКЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

1.1 Обзор существующих приложений и технологий

Одним из приложений, занимающим лидирующее положение в рейтинге, является «Фитнес: Тренировки Дома|в Зале». Пользователи оценили его более 4000 тысяч раз, а рейтинг приложения составляет 4,6 из 5 единиц.

Первом делом приложение собирает вводные данные, такие как возраст, вес, уровень активности, чтобы предоставить персонализированные тренировки. После этого пользователю предлагается главный экран, который уже заполнен упражнениями на основе предоставленных данных. Система составила персональный план тренировок, однако, пользователь может создать свою собственную тренировку, используя предоставленные упражнения, удобно разбитые по категориям. Каждое упражнение сопровождается текстовым описанием и видео с демонстрацией техники выполнения. Имеется возможность создать собственные упражнения фото и видео материалами. Упражнения, которые добавлены в тренировку, можно заполнить подходами. Имеется гибкая настройка единиц измерения. Приложение отличается приятной цветовой палитрой и единым стилем оформления во всех разделах приложения. Иконка выполнена в цветах приложения и содержит букву F, которая отсылает к названию. Интерфейс излишне перегружен, а навигация непонятна. Разработчик стоит сделать приложение более интуитивным и дать пользователям подсказки для взаимодействия с приложением. Некоторый базовый функционал глубоко запрятан, хотя его можно реализовать проще и это бы повлияло на восприятие приложения и сэкономило некоторое количество времени. Интерфейс приложения, иконка и цветовая палитра приложения представлены на рисунке 1.

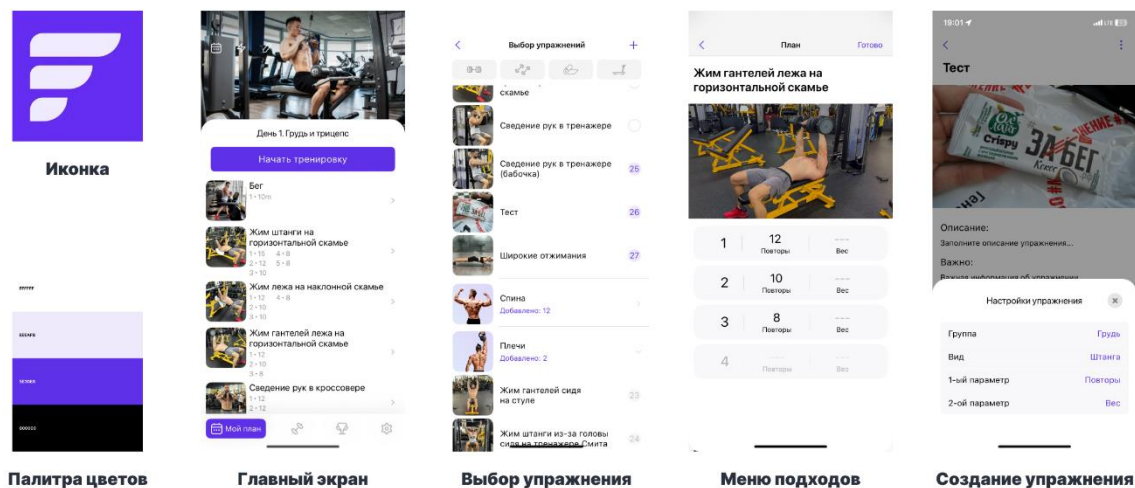


Рисунок 1. Дизайн приложения «Фитнес: Тренировки Дома|в Зале»

Приложение распространяется по условно-бесплатной модели с опциональной подпиской. Состав платной подписки следующий:

1. Отсутствие лимита на число записанных тренировок;
2. Персональный план тренировок;
3. Доступ к более чем 300 тренировкам на все группы мышц;
4. Доступ к истории тренировок и динамики весов с графиками;
5. Синхронизация с облачным сервисом iCloud, который обеспечивает синхронизацию между всеми устройствами экосистемы Apple.

В целом, приложение «Фитнес: Тренировки Дома|в Зале» имеет приятный дизайн, однако интерфейс и навигация сложны, а доступ к некоторым функциям затруднён. Базовые функции слишком глубоко запряты и неочевидны в использовании, что затрудняет взаимодействие с приложением. Стоит отметить наличие подписки, расширяющей базовый функционал приложения. Такая модель монетизации привлекательна для разработчиков, но для пользователей она выглядит как попытка заработать денег за доступ к незначительным функциям.

Другим популярным приложением является «Fitness Point: Дома и в Зале». Его рейтинг составляет 4,4 из 5 на основе более 1100 оценок пользователей.

При запуске приложения открывается меню с категориями упражнений. В каждой категории содержится множество упражнений, однако, большая часть из

них доступна по подписке. Тем не менее, имеется возможность добавлять собственные упражнения. При создании можно указать название и описание, выбрать группы мышц, участвующие в работе, и добавить фото для идентификации упражнения.

Решение с отображением работающих мышц не является хорошей идеей. Новичок не знает, какие мышцы участвуют в работе и не может их указать, а опытный спортсмен без всяких подсказок знает, какие мышцы работают. Эта функция совершенно бесполезна для опытных спортсменов. Возможно, эта функция полезна для предустановленных упражнений, чтобы облегчить старт для новичков. Приложение обладает разнообразным дополнительным функционалом: таймер, календарь тренировок, темная тема, настройка единиц измерения и выбор языка локализации. Однако, оно не выделяется своим дизайном на фоне остальных, она использует серую палитру цветов, которая ощущается устаревшей. На иконке ясно прослеживается силуэт Арнольда Шварценеггера, которой является иконой бодибилдинга 70-х годов. Иконка исполнена в характерных цветах приложения. Функции часто расположены в неочевидных местах, что создаёт отталкивающий пользовательский опыт. Основные экраны приложения, а также иконка и палитра цветов представлены на рисунке 2.

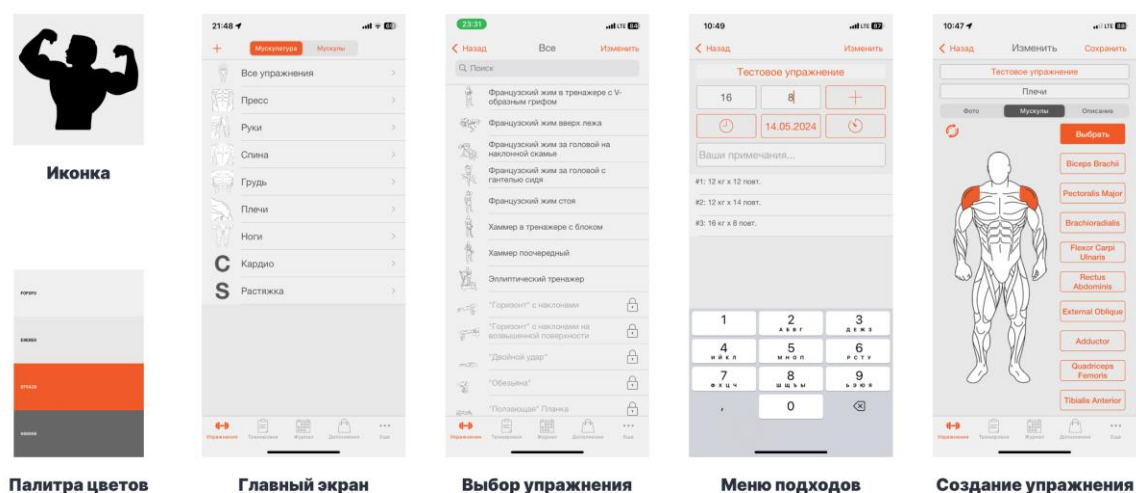


Рисунок 2. Дизайн приложения «Fitness Point: Дома и в Зале»

Приложение распространяется по условно-бесплатной модели, но с агрессивным навязыванием подписки и пейволлом (paywall), когда большая часть функционала доступна только по подписке. Вместе с рекламой в бесплатной версии создает ощущение, что разработчик заботится о монетизации больше, чем о качестве опыта пользователей. К платным функциям относится:

1. Доступ к более 400 упражнений;
2. График прогресса каждого упражнения;
3. Таймер для контроля отдыха между подходами;
4. Неограниченное количество записей в журнал. В бесплатной версии доступно всего 6 записей;
5. Отсутствие рекламы.

За дополнительную плату к стандартной подписки имеется возможность приобрести членство в Эксперт-клубе. Оно предоставляет доступ к большему количеству упражнений, к персональным тренировочным планам и синхронизацию с iCloud.

В общем, приложение предлагает хороший набор функций для отслеживать тренировок с большим разнообразием упражнений. Однако, простой интерфейс с непонятной навигацией и агрессивным навязыванием подписки ухудшают пользовательский опыт и побуждают пользователей обратить внимание на другие приложения.

Следующим приложением для анализа является «GymBook - Силовой Тренинг». Оно имеет рейтинг 4,6 на основе 102 рецензий. Цветовая гамма приятна глазу. При создании иконки использовался один из акцентных цветов приложения, гантель явно относит приложение к категории спортивных. Интерфейс схож с многими конкурентам, что не выделяет его среди остальных.

Разработчик предусмотрел, что с использованием его приложения могут возникнуть проблемы, поэтому добавил подсказки в раздел с тренировками и упражнениями. Чтобы проблем не возникало, нужно делать интуитивно понятные интерфейсы и продуманную навигацию.

Приложение открывается на главном экране, где расположены тренировки. Для добавления новой тренировки нужно нажать на иконку карандаша и только после этого появится возможность добавить тренировку через иконку плюса. Этот процесс неудобен и лишён смысла. Это лишнее нажатие на экран. Убрав иконку карандаш, функционал останется прежним, но пользовательский опыт улучшится. То же самое касается добавления и изменения упражнений. Приложение также испытывает проблемы с навигацией, однако, разработчик добавил подсказки для облегчения её понимания. Имеется возможность создать свое упражнение, с привычными параметрами для ввода. Проблема с отображением мышц аналогична приложению «Fitness Point: Дома и в Зале». Эта информация становится не нужной после первой тренировки. Разработчик заложил неверные смыслы в привычные функции: совершенно непонятно, как добавить новые подходы к упражнению. Сначала нужно заполнить подходы, затем выделить, и только после этого они становятся доступными для создания нового подхода в тренировке. Эта функция излишне сложная, и для упрощения достаточно убрать этап с выделением подходов и сразу добавлять подход в тренировку. Интерфейс приложения, его палитра цветов и иконка представлены на рисунке 3.

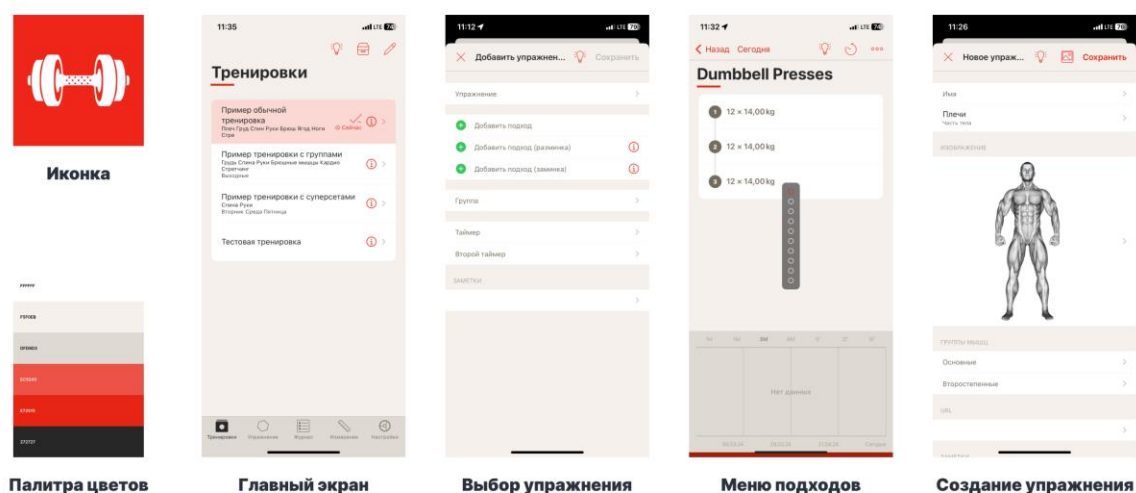


Рисунок 3. Дизайн приложения «GymBook ▪ Силовой Тренинг»

Приложение условно-бесплатное, но не ограничивает количество тренировок и упражнений. В состав платной подписки входят:

1. Дополнительные упражнения;
2. Синхронизация с iCloud;
3. Таймер для учёта времени отдыха и выполнения подходов;
4. Наборы тем и иконок.

В целом, приложение обладает потенциалом благодаря широкому набору функций, но для полного раскрытия этого потенциала необходимы доработки, направленные на улучшение пользовательского опыта и упрощение взаимодействия с приложением.

Приложение «Dropset: Gym tracker» выделяется среди остальных. Использование тёмных контрастных цветов в приложении подчёркивает его современный вид и элегантность. Минималистичный дизайн подчёркивает функциональность приложения, делая его лёгким в использовании и удобным для пользователей. Это отличный пример стильного и функционального дизайна, однако, иконка приложения не вызывает ассоциаций, связанных со спортом.

Главный экран приложения отображает созданные пользователем тренировки, которые затем заполняются упражнениями. Хотя упражнения уже заготовлены, пользователь всегда может создать свои. Взаимодействие пользователя с приложением происходит через взаимодействие с уникальными элементами, которые выделяют его на фоне остальных. Иконка, палитра цветов и интерфейс приложения демонстрируются на рисунке 4

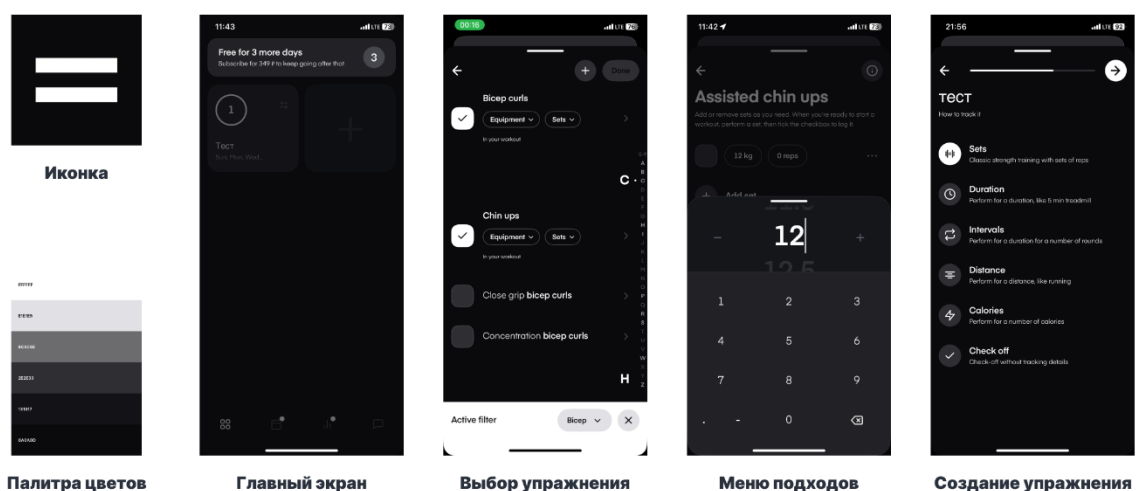


Рисунок 4. Дизайн приложения «Dropset: Gym tracker»

Функционал не отстаёт от дизайна, представляя понятные и удобные инструменты, подчёркнутые эффектными анимациями. Помимо этого, доступна статистика, календарь и глобальный чат для общения. Приложение распространяется через условно-бесплатную модель, но доступ к бесплатной версии ограничен тремя днями. После этого для продолжения использования нужно приобрести подписку.

Когда дизайн и функционал гармонично сочетаются, это обеспечивает удовлетворение пользовательских потребностей как в плане эстетики, так и в плане практичности.

Последнем в списке идёт приложение «Basic Gym». Оно полностью бесплатное, на этом преимущества заканчиваются.

На главном экране представлены категории упражнений, но они пусты. Пользователю необходимо заполнять их самостоятельно. Базовые функции, такие как добавление упражнений и подходов, реализованы неудачно. Отсутствуют инструменты для планирования тренировок. Кроме того, невзрачный дизайн усиливает негативное впечатление. Иконка приложения исполнена в цветах приложения и вызывает стойки ассоциации со спортом. Основные элементы, а также палитра цветов и иконка представлены на рисунке 5.

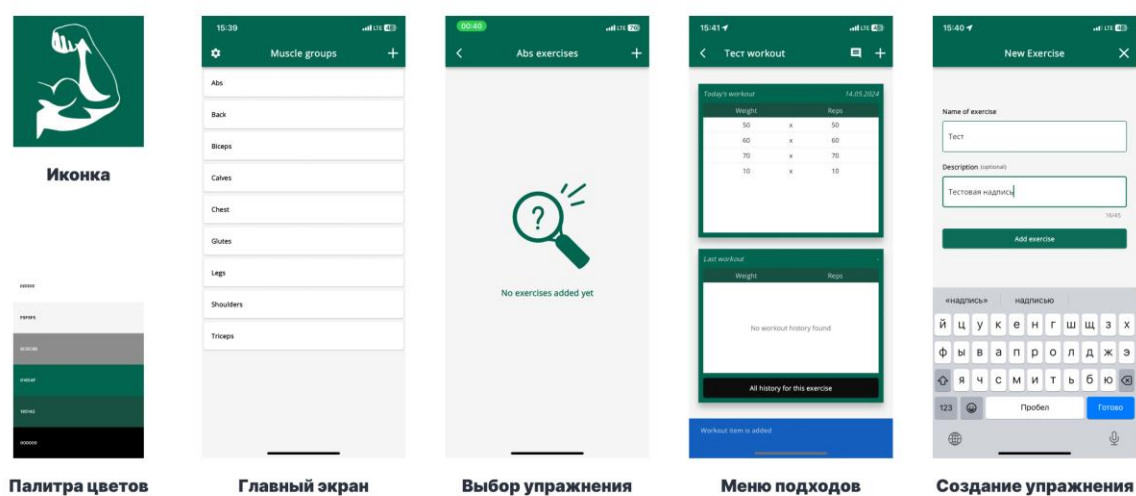


Рисунок 5. Дизайн приложения «Фитнес: Тренировки Дома|в Зале»

«Basic Gym» не соответствует ожиданиям пользователей и не выдерживает конкуренции с другими приложениями данной категории.

Анализируя различные мобильные приложения для тренировок, становится ясно, что на рынке наблюдается постоянное развитие с появлением разнообразных проектов. Высокий уровень конкуренции на рынке мобильных приложений заставляет разработчиков стремиться к соответствию ожиданиям пользователей. Приложения, не удовлетворяющие эти ожидания, рискуют потерять пользователей в пользу более качественных альтернатив. Большинство приложений используют условно-бесплатную модель монетизации, что подчёркивает важность нахождения баланса между заработком и пользовательским опытом. Гармоничное сочетание дизайна и функционала является ключом к удовлетворению потребностей пользователей.

В контексте мобильных приложений стоит отметить, что на этом рынке существует множество операционных систем, каждая из которых предоставляет разработчикам свои уникальные возможности. Однако две основные платформы, iOS и Android, выделяются из общего многообразия. Они доминируют на рынке мобильных операционных систем, соперничая друг с другом за внимание разработчиков и пользователей. iOS, разработанный компанией Apple, отличается высокой степенью оптимизации и стабильностью работы, а Android, в свою очередь, как открытая платформа, предлагает широкие возможности для настройки и адаптации приложений под различные устройства и потребности пользователей.

Решение в пользу iOS было принято из-за простоты оптимизации приложения под устройства только одного производителя - Apple, в отличие от Android, где требуется адаптация к широкому спектру устройств разных производителей. Этот выбор позволяет сосредоточить усилия на создании качественного пользовательского опыта, а также обеспечить более простую поддержку и обновление приложения.

Выбор технологии играет ключевую роль в разработке и влияет на многие аспекты, включая функциональность, производительность, безопасность и доступные методы взаимодействия с пользователем. От правильного выбора

зависит успех проекта, поэтому необходимо провести тщательный анализ инструментов iOS-разработки. Можно выделить несколько ключевых фреймворков, каждый из которых имеет свои особенности и преимущества.

SwiftUI, представленный Apple в 2019 году, является современным фреймворком для разработки пользовательских интерфейсов. Логотип данной технологии представлен на рисунке 6. Среди его ключевых преимуществ можно выделить удобство написания кода и скорость верстки, активную поддержку от Apple и широкое сообщество разработчиков. Кроме того, SwiftUI предлагает возможность предварительного просмотра в реальном времени, что значительно упрощает процесс разработки. Совместимость с UIKit позволяет интегрировать компоненты этих фреймворков друг в друга. Декларативный подход SwiftUI делает код более простым и читаемым, а кроссплатформенность внутри экосистемы Apple позволяет создавать приложения для iOS, macOS, watchOS, tvOS и visionOS. Важной особенностью является также поддержка нативных баз данных CoreData и SwiftData.



Рисунок 6. Логотип фреймворка SwiftUI

Однако, у SwiftUI есть и недостатки. Он поддерживается только на iOS 13 и более поздних версиях, что исключает устройства с более старыми версиями

операционной системы. Кроме того, частые обновления и изменения в SwiftUI могут создавать проблемы с совместимостью старых и новых методов. Несмотря на эти минусы, многочисленные преимущества делают SwiftUI отличным выбором для разработки приложений.

UIKit, использующийся с 2007 года, продолжает оставаться надежным фреймворком для создания пользовательских интерфейсов в iOS-приложениях. Его главные плюсы включают стабильность, высокую производительность и надежность. UIKit предоставляет полный контроль над интерфейсом и совместим с более ранними версиями iOS, что позволяет разрабатывать приложения для широкой аудитории. Он также поддерживается множеством сторонних библиотек и инструментов.

Тем не менее, UIKit имеет свои недостатки. Он требует написания более сложного и детализированного кода, что замедляет процесс разработки. В будущем UIKit отойдет на второй план и акцент будет смещен на SwiftUI и его современный подход к разработке.

Другим популярным фреймворком с открытым исходным кодом является React Native. Логотип React Native представлен на рисунке 7. Его главные достоинства включают использование JavaScript, активное сообщество разработчиков и возможность создания приложений, работающих как на iOS, так и на Android. Это обеспечивает единообразный интерфейс и пользовательский опыт на всех платформах.

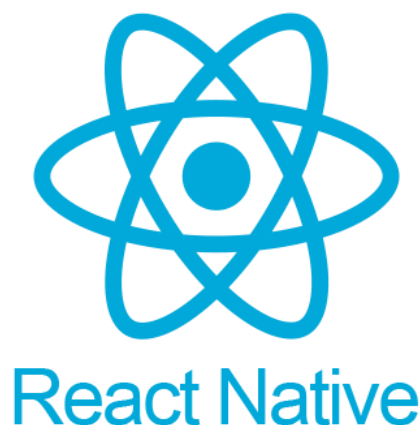


Рисунок 7. Логотип фреймворка React Native

Однако, React Native не безупречен. Возникают трудности при развертывании приложений для iOS, частыми обновлениями, которые могут нарушать совместимость, а также с низкой производительностью и отсутствием визуального редактора интерфейсов. Несмотря на эти проблемы, React Native остаётся хорошим выбором для кроссплатформенной разработки, но для приложений, ориентированных на устройства Apple, лучше подойдут нативные фреймворки от Apple.

Среди различных фреймворков SwiftUI оказался наиболее привлекательным вариантом для создания мобильных приложений в экосистеме Apple. Это новейший фреймворк компании Apple, и в настоящее время акцент делается именно на нём. Компания активно развивает его, формируется база разработчиков и профессиональное сообщество. SwiftUI пришёл на смену UIKit, который долгое время являлся основным инструментом разработки приложений в экосистеме Apple. Все будущие приложения будут разрабатываться с использованием SwiftUI, а существующие приложения, написанные на UIKit, будут постепенно переходить на новый фреймворк. Выбор SwiftUI представляется наиболее обоснованным и перспективным с точки зрения скорости разработки, эффективности, совместимости и будущего развития.

1.2 Требования к мобильному приложению

Анализ мобильных приложений для учёта тренировок выявил как положительные, так и отрицательные примеры на рынке. Для того чтобы приложение стало популярным и нашло свою аудиторию, необходимо уделить внимание следующим аспектам:

1. Простота использования и интуитивно понятная навигация важны для пользователей, поэтому нужно отобразить тренировки пользователя на главном экране. Такая реализация обеспечит быстрый доступ к функционалу и приятный пользовательский опыт;

2. Приложение должно предоставлять базовый функционал, характерный для систем учёта тренировок: возможность создания, удаления, переименования тренировок, упражнений и подходов;
3. Важно учитывать разнообразие географии пользователей, предоставив выбор единиц измерения, таких как килограммы и фунты, и локализацию на основные международные языки;
4. Предоставление начального списка упражнений поможет пользователям быстрее начать работу с приложением и обезопасит разработчика от недопонимания со стороны аудитории;
5. Приложение должно быть полностью бесплатным. Такой подход формирует лояльность среди аудитории и допускает более широкий её охват;
6. Приложение должно отслеживать статистику упражнений и прогресс пользователей;
7. Все данные должны храниться локально с помощью нативной базы данных Core Data;
8. Приложение должно быть выполнено в едином стиле, сочетая дизайн и функциональность. Дизайн должен выделяться на фоне конкурентов, а иконка - узнаваемой и отражающей суть приложения.

Разработка приложения с опорой на эти требования позволит создать уникальное приложение для учёта тренировок, ориентированное на широкую аудиторию без привязки к определённому региону.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА МОБИЛЬНОГО УПРАЖНЕНИЯ

2.1 Прототип пользовательского интерфейса

Процесс проектирования пользовательского интерфейса мобильного приложения «Gym Tracker» начался с тщательного анализа потребностей и ожиданий целевой аудитории. Название в переводе с английского означает «Трекер для тренажерного зала», что отражает основную идею приложения. Основной задачей было создание простого и понятного интерфейса. Были определены основные пользовательские сценарии, описывающие типичные действия, которые пользователи будут выполнять в приложении. Эти сценарии включали:

- Управление тренировками: Пользователи должны иметь возможность добавлять новые тренировки, редактировать существующие и удалять их при необходимости.
- Управление упражнениями: Пользователи должны иметь возможность выбирать упражнения из списка, создавать собственные упражнения, редактировать существующие и удалять их при необходимости.
- Управление подходами: Пользователи должны иметь возможность добавлять и удалять подходы для упражнений.
- Просмотр статистики: Важно предоставить пользователям возможность детально просматривать статистику упражнений.

В итоге был разработан прототип приложения, который отражал типичный для систем учёта тренировок функционал и описанные пользовательские сценарии. Этот прототип создан инструментами Figma - популярного онлайн-сервиса, предназначенного для создания макетов, прототипов и интерфейсов.

Прототип разделен на несколько экранов, каждый из которых отражает определенные функции. На главном экране пользователь видит список своих тренировок с названиями, датами и днями недели. Пользователь может добавить

новую тренировку, указав название и дату в открывшемся окне. Также пользователь может редактировать или удалять существующие тренировки. Общее число тренировок отображается в нижней части экрана.

Экран настроек позволяет управлять списком доступных упражнений и инвентаря, выбирать единицу измерения веса и формулу для расчёта одного повторения с максимальным весом.

Экран списка упражнений предоставляет возможность группировать упражнения по категориям, менять порядок, а также добавлять новые упражнения или удалять существующие. У инвентаря схожий функционал.

Экран тренировки отображает список упражнений и предоставляет возможностью добавления новых упражнений или удаления существующих. Здесь также можно увидеть информацию о подходах, включая вес и число повторений. Пользователь может добавлять подходы с указанием веса и количества повторений, а также перемещать или удалять уже существующие подходы.

Экран статистики предоставляет информацию по упражнению. Она делится на информацию о текущей тренировке, информации по упражнению за всё время, и на информацию о максимальном возможном разе. Также доступно поле для заметок, чтобы описать свои ощущения от выполнения.

Цветовая схема приложения была выбрана с целью обеспечения визуального комфорта и эстетического удовольствия для пользователей. Основными цветами, используемыми в интерфейсе, являются различные оттенки синего и фиолетового, которые гармонично сочетаются друг с другом.

Использование градиента создаёт приятный визуальный эффект. Он не только делает приложение привлекательным, но и разграничивает элементы интерфейса, что делает навигацию по приложению интуитивно понятной и визуально приятной. Подход к выбору цветовой схемы учитывает функциональные и эстетические аспекты, что обеспечивает гармоничное восприятие приложения. Прототип дизайна мобильного приложения «Gym Tracker» демонстрируется на рисунке 8.

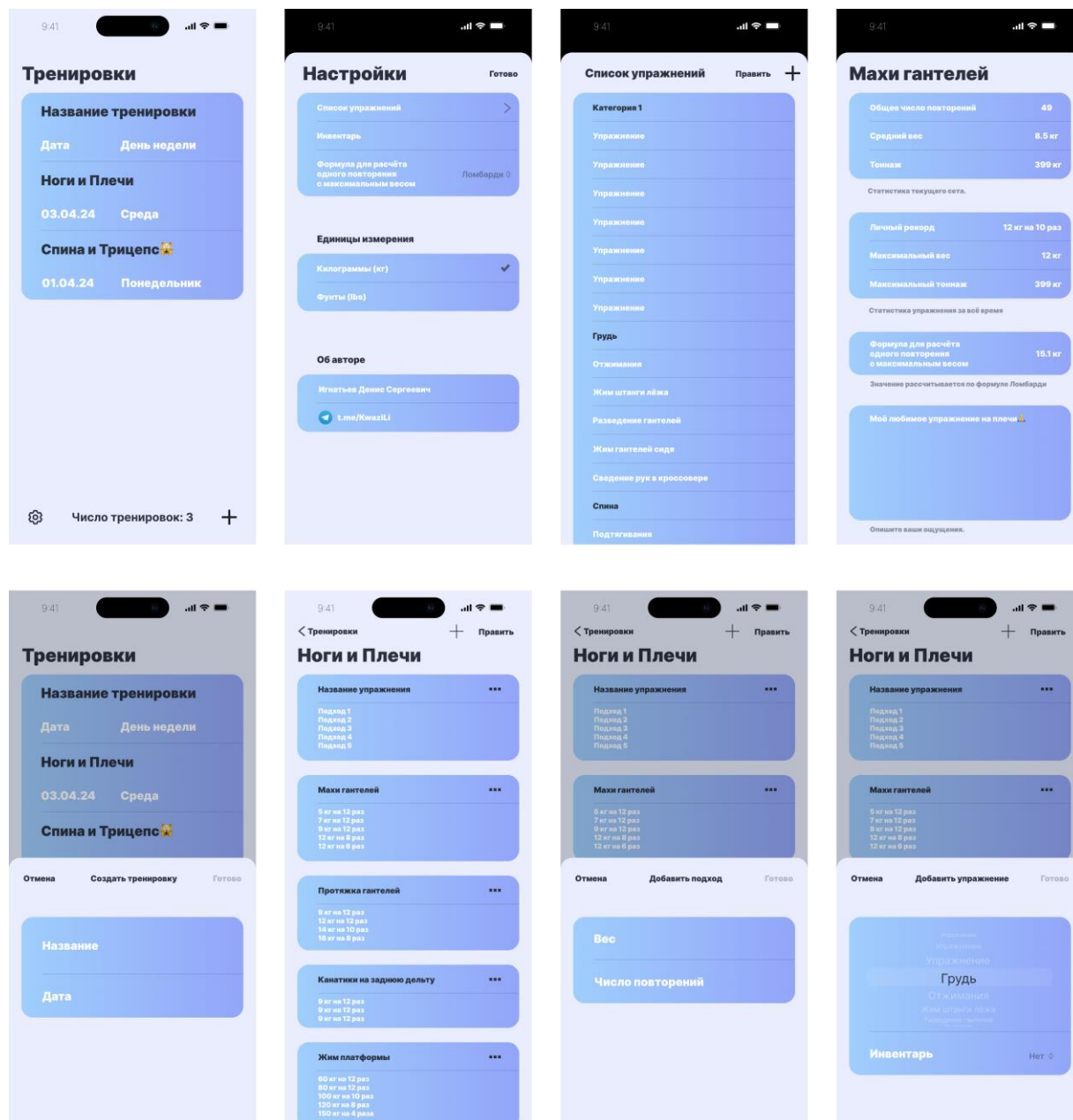


Рисунок 8. Прототип дизайна приложения «Gym Tracker»

Такой дизайн выделяет «Gym Tracker» среди конкурентов, делая его не только удобным и эффективным инструментом для отслеживания тренировок, но и приятным в использовании продуктом.

2.2 Создание иконки

Процесс создания иконки для мобильного приложения является важной частью проектирования, так как иконка является первым визуальным элементом, с

которым взаимодействует пользователь. Она должна быть узнаваемой, запоминающейся и отражать основную идею и функционал приложения.

Был проведен анализ иконок приложений для учёта тренировок, чтобы определить успешные и неудачные элементы в их дизайне. Многие иконки конкурентов не оставляют запоминающегося впечатления и часто представляют собой стереотипные изображения, которые сложно отличить друг от друга. Некоторые вовсе не связаны с тематикой приложения. Такие иконки теряются среди множества конкурентов, не имея уникальных отличительных черт. Рассматриваемые иконки представлены на рисунке 9.

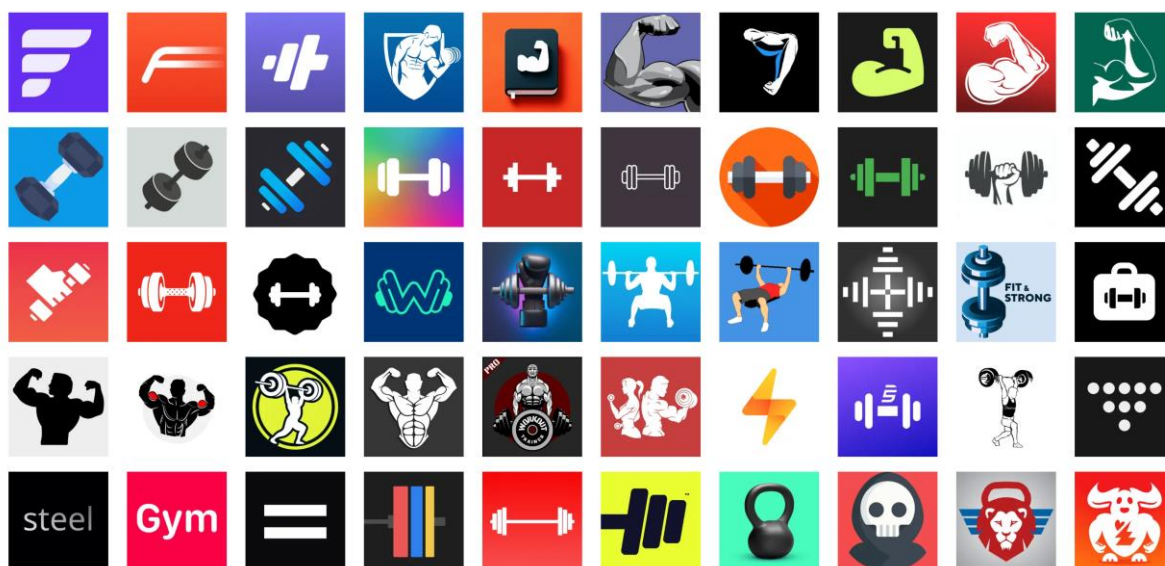


Рисунок 8. Дизайн иконок существующих приложений

Цветовая палитра большинства иконок включает цвета, используемые в интерфейсе приложения. Часто используются яркие, но плохо гармонирующие цвета. Такие цветовые решения не привлекают внимание, но и отталкивают потенциальных пользователей.

Среди рассмотренных программных является непопулярным решением использовать маскотов. Они могут значительно повысить уникальность бренда и сделать приложение более привлекательным и запоминающимся. Успешным примером использования маскота является приложение «Duolingo», где персонаж

сова стала символом приложения и важной частью взаимодействия с пользователем. Сова-маскот изображена на рисунке 9.



Рисунок 9. Маскот приложения «Duolingo»

Включение маскота в дизайн иконки выделяет продукт среди конкурентов и повышает узнаваемость среди пользователей.

В целом, иконки конкурентов в основном используют стандартные фитнес-символы и яркие контрастные цвета для привлечения внимания. Многие из них не обладают уникальностью, которая бы выделяла их среди множества других приложений для отслеживания тренировок.

Существует множество подходов для создания иконок. Иконку можно нарисовать вручную или воспользоваться услугами профессионального дизайнера. Однако, в последнее время популярность набирают сервисы для генерации изображений, которые значительно ускоряют процесс создания изображений, обеспечивают высокий уровень качества и дают возможность творить людям без определённых навыков. Для создания иконки приложения «Gym Tracker» была выбрана генеративная нейронная сеть DALL·E 3 от компании OpenAI, известная своим точным пониманием запросов и высоким качеством изображений.

Иконка приложения создана по следующему запросу: «Tiger mascot, dumbbells in his hands, icon, minimalism, Flat style, 3D Rendering, simple shapes», что переводится как: «Талисман-тигр, гантели в его руках, иконка, минимализм,

плоский стиль, 3D-рендеринг, простые формы». В результате было создано изображение милого тигра с гантелями в лапах.

Для дальнейшего улучшения иконки изображение было доработано с помощью нейронной сети Stable Diffusion. Эта генеративная нейронная сеть позволила добавить дополнительные детали, дорисовать гантели и устранить артефакты, улучшив общее качество изображения. Затем с помощью программы Adobe Photoshop фон изображения был изменен, чтобы соответствовать цветовой палитре приложения, а также маскот был размещен по центру. Дополнительно, с помощью Stable Diffusion, была заменена майка тигра на более подходящую по стилю. Процесс создания иконки представлен на рисунке 10.

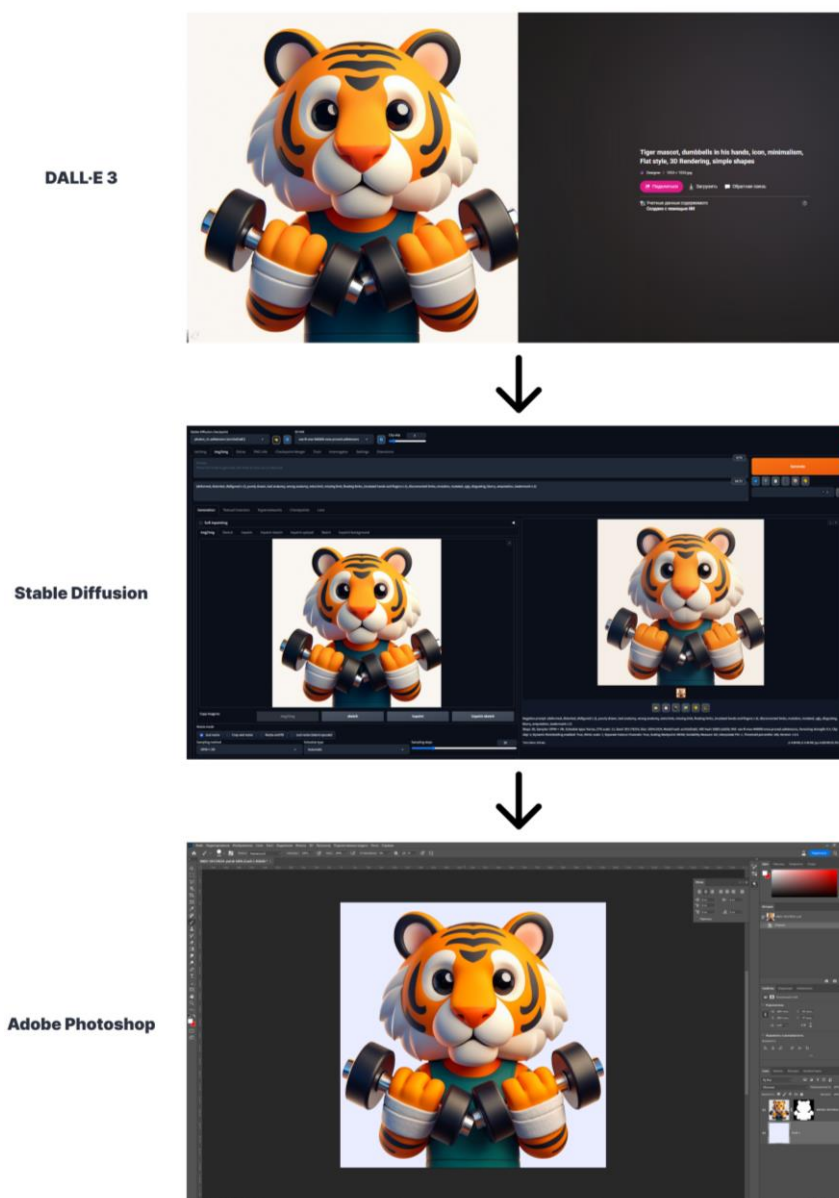


Рисунок 10. Процесс создание иконки

В итоге иконка приложения «Gym Tracker» представляет собой изображение милого тигра, держащего гантели в обеих лапах. Тигр одет в белую майку, а на его запястьях напульсники, что подчеркивает спортивную тематику. Лицо тигра изображено с большими глазами и дружелюбным выражением вызывает положительные эмоции и привлекает внимание пользователей. Такая иконка не только символизирует основную концепцию приложения, но и добавляет элемент дружелюбности и запоминаемости. Финальный вариант иконки приложения «Gym Tracker» представлен на рисунке 11.



Рисунок 11. Иконка приложения «Gym Tracker»

Уникальный маскот делает иконку легко узнаваемой, позволяя пользователям быстро находить приложение среди других. Дизайн иконки хорошо масштабируется на различные размеры экранов и устройств, сохраняя четкость и привлекательность. Даже в уменьшенном виде основные элементы остаются различимыми и узнаваемыми.

Иконка приложения «Gym Tracker» выделяется на фоне конкурентов благодаря своему уникальному дизайну, гармоничной цветовой палитре и дружелюбному персонажу. Она отражает функционал приложения и легко узнаваема, эмоционально привлекательна и хорошо масштабируема.

2.3 Описание процесса разработки

Разработка приложения осуществлялась в среде разработки программного обеспечения Xcode, которая является мощным инструментом для создания приложений под платформы Apple. Xcode предоставляет разработчикам широкий набор инструментов и возможностей для создания, тестирования и оптимизации приложений. Данные пользователей хранятся локально с использованием нативной базы данных Core Data.

Итоговый вариант структуры проекта представлен следующими папками и файлами, каждый из которых имеет своё назначение. Структура проекта представлена на рисунке 12.

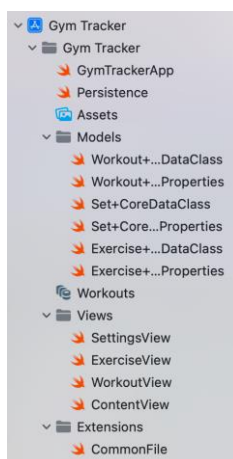


Рисунок 12. Структура проекта

Модель данных приложения описывается тремя сущностями, каждая из которых имеет свою уникальную роль и набор атрибутов, необходимых для корректного функционирования приложения:

Сущность **Workout** представляет собой данные о конкретной тренировке. Она содержит атрибуты, такие как дата (**date**) и название тренировки (**name**). Кроме того, данная сущность имеет связь с упражнениями (**exercises**), которые выполняются в рамках данной тренировки. Одна тренировка может включать множество упражнений (связь «один ко многим»).

Сущность **Exercise** описывает данные об упражнениях. Она содержит атрибуты, такие как название упражнения (**name**), используемое оборудование (**equipment**) и заметки, связанные с упражнением (**notes**). **Exercise** имеет связи с тренировкой, в которой оно выполняется (**workout**), и с подходами (**sets**). Одно упражнение может иметь множество подходов (связь «один ко многим»).

Сущность **Set** описывает данные о конкретном подходе для упражнения. Она содержит такие атрибуты, как количество повторений (**reps**) и вес (**weight**). Каждый подход связан с определенным упражнением (**exercise**).

Модель данных приложения «Gym Tracker» представлена на рисунке 13.

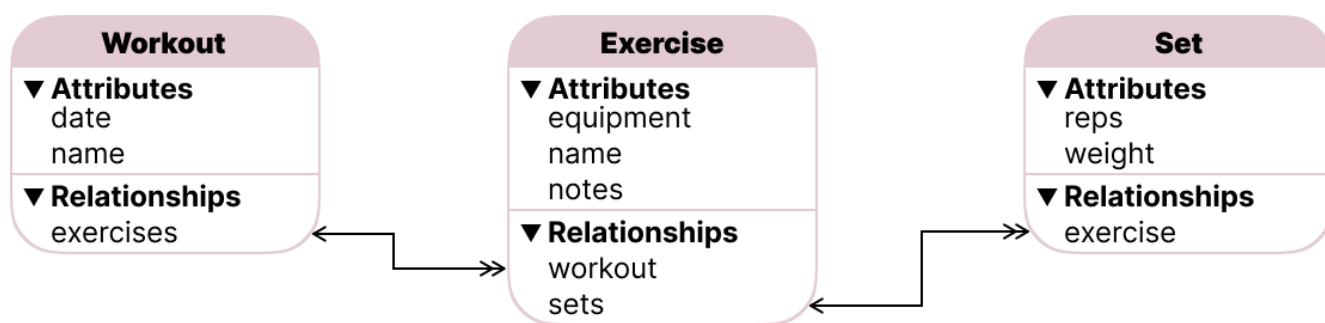
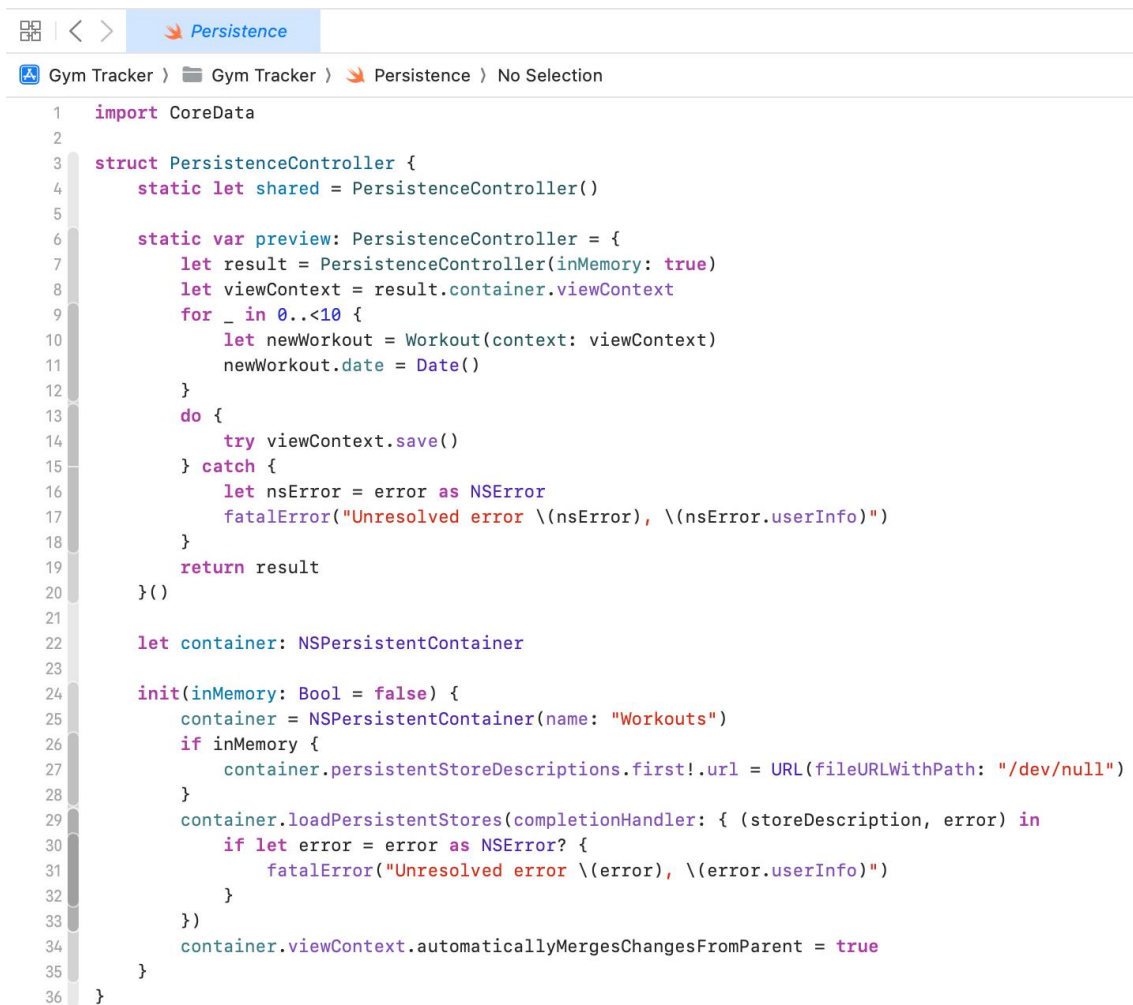


Рисунок 13. Модель данных

Эти сущности полностью реализуют функционал приложения для учёта тренировок и позволяют манипулировать тренировками, упражнениями и их подходами.

Файл Persistence представляет собой структуру, обеспечивающую централизованный доступ к системе управления постоянством данных Core Data. Код представлен на рисунке 14.



```
1 import CoreData
2
3 struct PersistenceController {
4     static let shared = PersistenceController()
5
6     static var preview: PersistenceController = {
7         let result = PersistenceController(inMemory: true)
8         let viewContext = result.container.viewContext
9         for _ in 0..<10 {
10             let newWorkout = Workout(context: viewContext)
11             newWorkout.date = Date()
12         }
13         do {
14             try viewContext.save()
15         } catch {
16             let nsError = error as NSError
17             fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
18         }
19         return result
20     }()
21
22     let container: NSPersistentContainer
23
24     init(inMemory: Bool = false) {
25         container = NSPersistentContainer(name: "Workouts")
26         if inMemory {
27             container.persistentStoreDescriptions.first!.url = URL(fileURLWithPath: "/dev/null")
28         }
29         container.loadPersistentStores(completionHandler: { (storeDescription, error) in
30             if let error = error as NSError? {
31                 fatalError("Unresolved error \(error), \(error.userInfo)")
32             }
33         })
34         container.viewContext.automaticallyMergesChangesFromParent = true
35     }
36 }
```

Рисунок 14. Файл Persistence

Assets представляет собой каталог с ресурсами, в котором хранятся все изображения, иконки и цвета, используемые в приложении. Они представлены на рисунке 15.

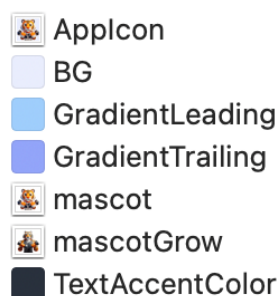
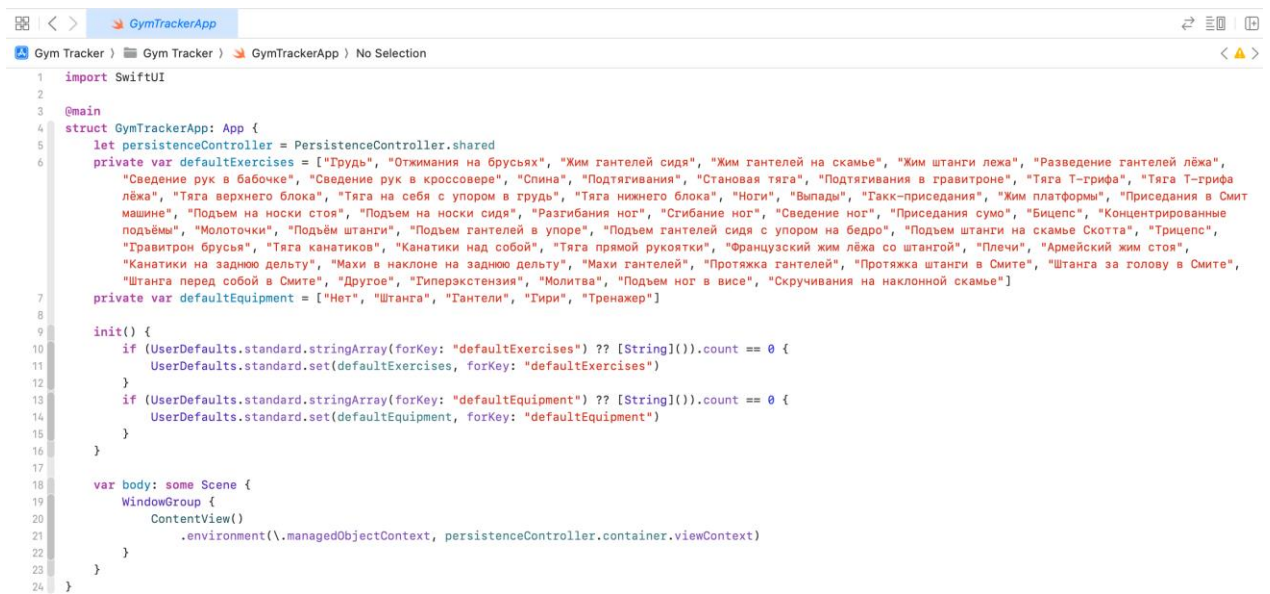


Рисунок 15. Ресурсы, используемые в приложении

GymTrackerApp является точкой входа в приложение. Этот код обеспечивает инициализацию и настройку приложения, устанавливая данные по умолчанию и передавая контекст Core Data для управления данными. Переменные defaultExercises и defaultEquipment содержат перечень упражнений и оборудования по умолчанию, что позволяет пользователю сразу начать работу с приложением. Код представлен на рисунке 16.



```
1 import SwiftUI
2
3 @main
4 struct GymTrackerApp: App {
5     let persistenceController = PersistenceController.shared
6     private var defaultExercises = ["Грудь", "Отжимания на брусьях", "Жим гантелей сидя", "Жим гантелей на скамье", "Жим штанги лежа", "Разведение гантелей лёжа",
7         "Сведение рук в бабочке", "Сведение рук в кроссовере", "Спина", "Подтягивания", "Становая тяга", "Подтягивания в гравитроне", "Тяга Т-грифа", "Тяга Т-грифа лёжа",
8         "Тяга верхнего блока", "Тяга на себя с упором в грудь", "Тяга нижнего блока", "Ноги", "Выпады", "Гакк-приседания", "Жим платформы", "Приседания в Смит машине",
9         "Подъем на носки стоя", "Подъем на носки сидя", "Разгибания ног", "Сгибание ног", "Сведение ног", "Приседания сумо", "Бицепс", "Концентрированные подъемы",
10        "Молоточки", "Подъем штанги", "Подъем гантелей в упоре", "Подъем гантелей сидя с упором на бедро", "Подъем штанги на скамье Скотта", "Трицепс",
11        "Гравитрон брусья", "Тяга канатиков", "Канатки над собой", "Тяга прямой рукоятки", "Французский жим лёжа со штангой", "Плечи", "Армейский жим стоя",
12        "Канатки на заднюю дельту", "Махи в наклоне на заднюю дельту", "Махи гантелей", "Протяжка гантелей", "Протяжка штанги в Смите", "Штанга за голову в Смите",
13        "Штанга перед собой в Смите", "Другое", "Гиперэкстензия", "Молитва", "Подъем ног в висе", "Скручивания на наклонной скамье"]
14    private var defaultEquipment = ["Нет", "Штанга", "Гантели", "Гири", "Тренажер"]
15
16     init() {
17         if (UserDefaults.standard.stringArray(forKey: "defaultExercises") ?? [String]()).count == 0 {
18             UserDefaults.standard.set(defaultExercises, forKey: "defaultExercises")
19         }
20         if (UserDefaults.standard.stringArray(forKey: "defaultEquipment") ?? [String]()).count == 0 {
21             UserDefaults.standard.set(defaultEquipment, forKey: "defaultEquipment")
22         }
23     }
24
25     var body: some Scene {
26         WindowGroup {
27             ContentView()
28             .environment(\.managedObjectContext, persistenceController.container.viewContext)
29         }
30     }
31 }
```

Рисунок 16. Файл GymTrackerApp

Файл CommonFile расширяет функциональность существующих классов и структур. Расширения позволяют применять стили ко многим компонентам пользовательского интерфейса, обеспечивая лаконичность кода и единообразие внешнего вида приложения. Реализованные функции позволяют задать градиент, установить задний фон всего приложения, изменить стандартный стиль текста, секций, футера (footer) и хедера (header), кнопок, заголовков и текстовых полей.

Пример реализации функций градиента, фона приложения и стиля текста представлены на рисунке 17.

```
func accentGradient() -> some View {
    self.foregroundColor(LinearGradient(colors: [.gradientLeading, .gradientTrailing], startPoint: .leading, endPoint: .trailing))
}
func bgColor() -> some View {
    Color("BG").ignoresSafeArea()
}
func textAccentColor() -> some View {
    self.foregroundColor(Color("TextAccentColor"))
}
```

Рисунок 17. Функции градиента, фона приложения и стиля текста

ContentView представляет главный экран приложения, отображающий список тренировок. С этого экрана начинается взаимодействие пользователя с приложением. Реализована возможность добавлять новые тренировки, переименовывать их, а также удалять. Код, отражающий функционал, представлен на рисунке 18.

```
private func addWorkout() {
    withAnimation {
        let newWorkout = Workout(context: viewContext)
        newWorkout.date = newDate
        newWorkout.name = newName
        newWorkout.exercises = []

        isShowingSheet = false
        newDate = Date.now
        newName = ""

        do {
            try viewContext.save()
        } catch {
            let nsError = error as NSError
            fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
        }

        multiSelection = Swift.Set<Workout>()
    }
}

private func renameWorkout() {
    guard let workout = currentWorkout else { return }

    withAnimation {
        workout.name = newName
        workout.date = newDate

        do {
            try viewContext.save()
        } catch {
            let nsError = error as NSError
            fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
        }

        isShowingRenameSheet = false
        newName = ""
        newDate = Date.now
        currentWorkout = nil
    }
}

private func deleteWorkout(workout: Workout) {
    withAnimation {
        viewContext.delete(workout)

        do {
            try viewContext.save()
        } catch {
            let nsError = error as NSError
            fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
        }

        multiSelection = Swift.Set<Workout>()
    }
}
```

Рисунок 18. Функции создания, переименования и удаления тренировок

В данном коде реализуется навигационное представление (NavigationView) с использованием списка (List), где отображаются тренировки. Каждая тренировка представлена в виде элемента списка, содержащего номер тренировки, ее название, дату и день недели. Каждый элемент списка имеет контекстное меню, где можно выбрать действия «Удалить» или «Переименовать» выбранную тренировку. Кроме того, элементы списка имеют возможность свайпа вправо для удаления и влево для переименования тренировки. Навигация и функционал отражены на рисунке 19.

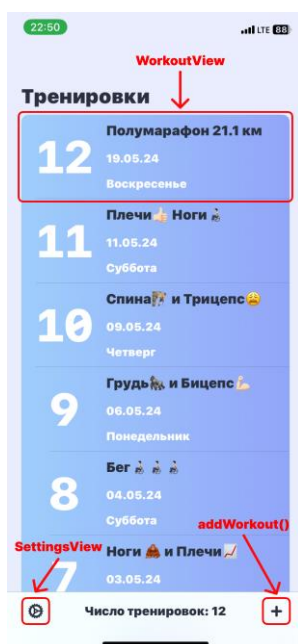


Рисунок 19. Функционал и навигация главного экрана приложения

WorkoutView реализует функциональность экрана тренировки, отображая список упражнений, связанных с текущей тренировкой. Пользователь имеет возможность добавлять новые упражнения, вызывая модальное окно (sheet) нажатием кнопки «plus». Функция добавления нового упражнения в тренировку представлена на рисунке 20.

```
private func addExercise() {
    let newExercise = Exercise(context: viewContext)
    newExercise.name = newName
    newExercise.equipment = newEquipment

    workout.addToExercises(newExercise)

    isShowingSheet = false
    newName = ""

    do {
        try viewContext.save()
    } catch {
        let nsError = error as NSError
        fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
    }
}
```

Рисунок 20. Функция добавления нового упражнения

В модальном окне можно выбрать упражнение из predetermined списка или ввести название самостоятельно, если эта функция активна в настройках. Также предоставляется выбор инвентаря из доступных вариантов и возможность удалить упражнение из тренировки.

ExerciseView включает в себя различные элементы пользовательского интерфейса для управления упражнениями, такие как добавление подходов и изменение упражнения. Код, отражающий управление подходами, представлен на рисунке 21.

```
private func addSet() {
    withAnimation {
        let newSet = Set(context: viewContext)
        newSet.reps = Int16(newReps) ?? 0 <= 0 ? 1 : (Int16(newReps) ?? 1)
        newSet.weight = Float(newWeight) ?? 0 <= 0 ? 1 : (Float(newWeight) ?? 1)

        exercise.addToSets(newSet)

        isShowingSetSheet = false
        newReps = ""
        newWeight = ""

        do {
            try viewContext.save()
        } catch {
            let nsError = error as NSError
            fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
        }
    }
}

private func deleteSet(offsets: IndexSet) {
    withAnimation {
        if let sets = exercise.sets.array as? [Set] {
            offsets.map { sets[$0] }.forEach(viewContext.delete)

            do {
                try viewContext.save()
            } catch {
                let nsError = error as NSError
                fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
            }
        }
    }
}

private func moveSet(fromOffsets indexSet: IndexSet, toOffset index: Int) {
    withAnimation {
        if let sets = exercise.sets.array as? [Set] {
            var mutableSets = sets
            mutableSets.move(fromOffsets: indexSet, toOffset: index)

            exercise.removeFromSets(exercise.sets)

            for set in mutableSets {
                exercise.addToSets(set)
            }

            do {
                try viewContext.save()
            } catch {
                let nsError = error as NSError
                fatalError("Unresolved error \(nsError), \(nsError.userInfo)")
            }
        }
    }
}
```

Рисунок 21. Функции добавления, удаления и перемещения подходов

Код также определяет структуру StatsView, которая представляет собой статистику по конкретному упражнению за текущую тренировку или за всё время.

Для перехода в экрану статистики упражнения используется `NavigationLink`. Экран статистики упражнения представлен на рисунке 22.



Рисунок 22. Экран статистики упражнения

SettingsView реализует функциональность экрана настроек, где пользователь может управлять предустановленными упражнениями, инвентарем, единицами измерения и формулой расчёта максимального веса для одного повторения. Меню настроек представлено на рисунке 23.

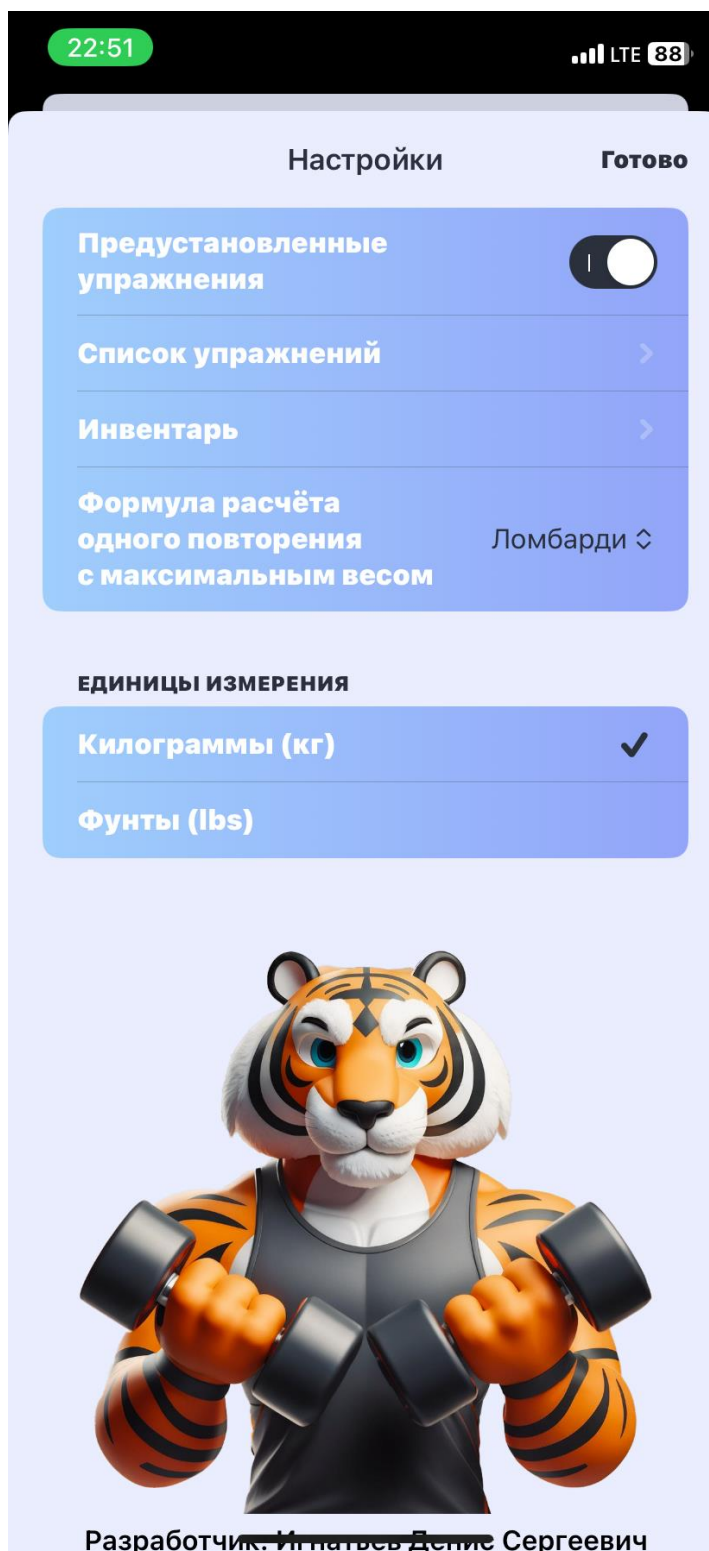


Рисунок 23. Меню настроек

Вложенная структура DefaultsView отвечает за отображение списка предустановленных элементов и их управление, предоставляя методы для добавления, удаления и перемещения элементов в списке предустановленных упражнений и инвентаря. Функциональность отображена на рисунке 24.

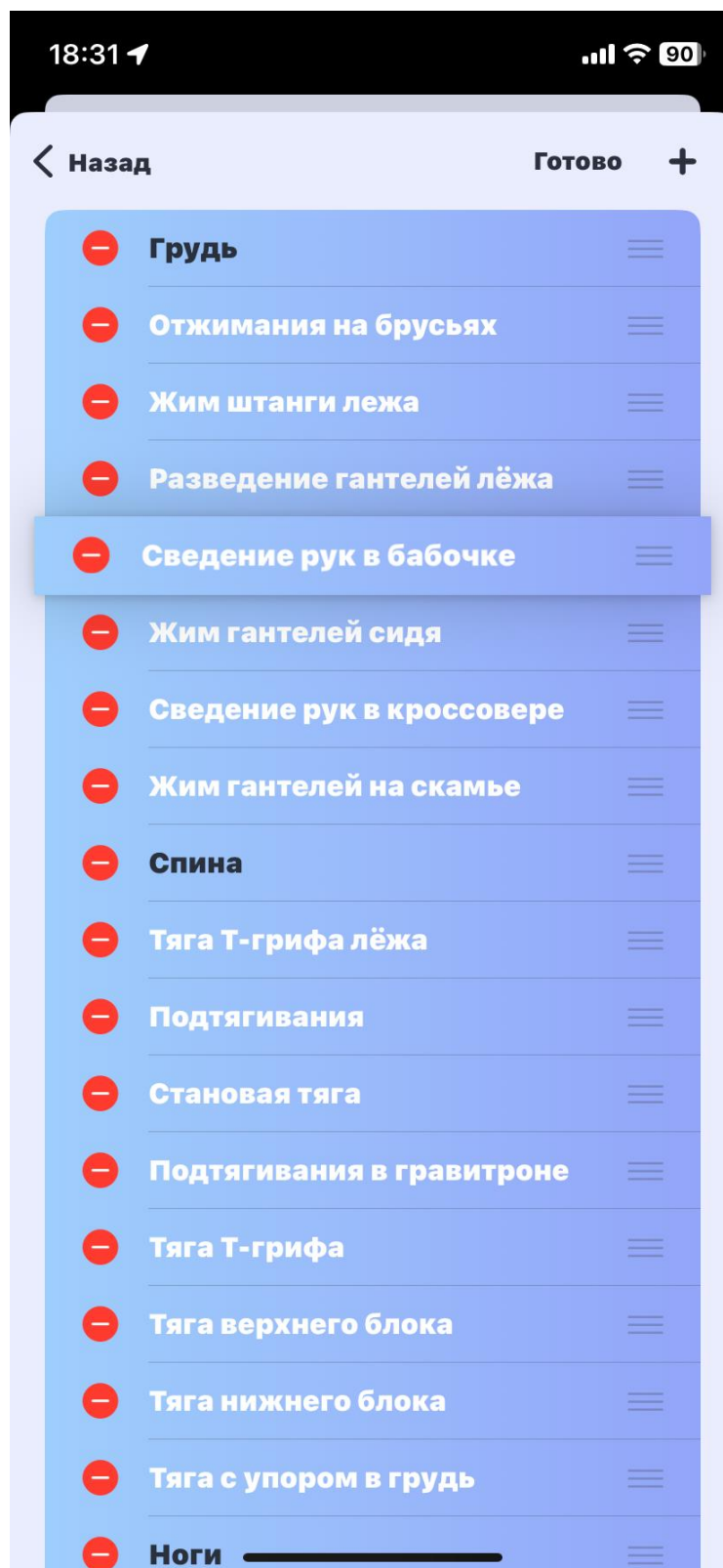


Рисунок 24. Функционал перемещения и удаления упражнений из списка

В результате работы реализовано приложение для учёта тренировок «Gym Tracker». Были учтены все требования, чтобы создать приложение, ориентированное на широкую аудиторию. Оно обладает приятным дизайном, понятной навигацией и широким функционалом, а его иконка выделяется на фоне остальных. Интерфейс приложения, иконка и палитра цветов представлены на рисунке 25.

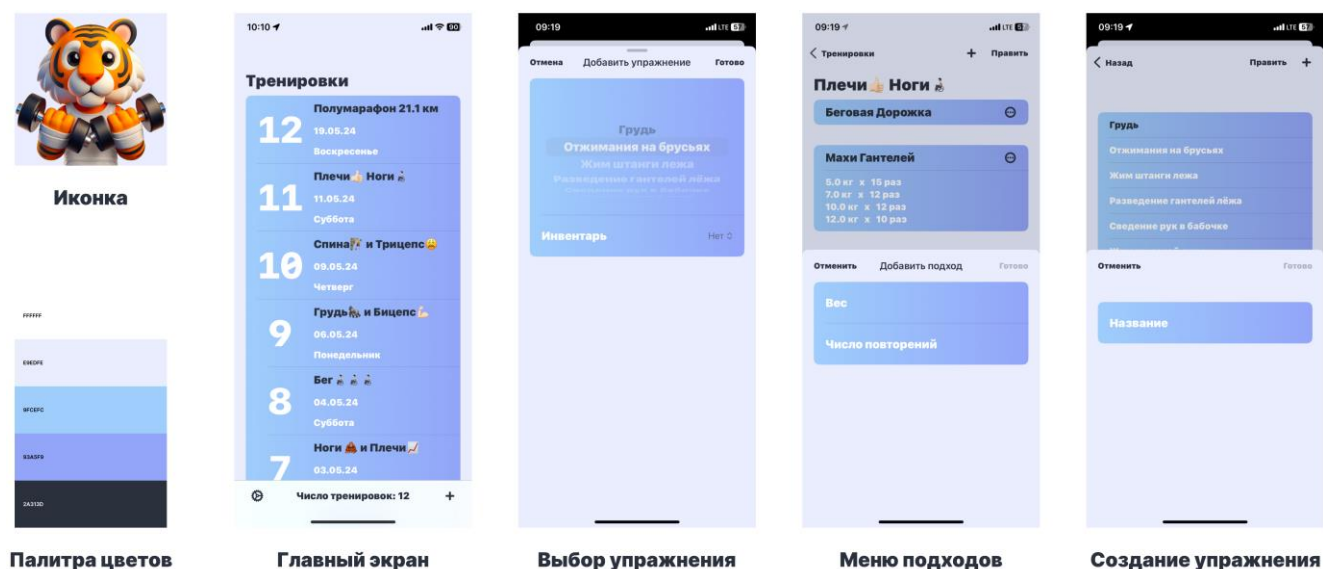


Рисунок 25. Дизайн приложения «Gym Tracker»

Все эти элементы собираются воедино и образуют завершённое приложение, готовое к запуску и дальнейшему развитию. Оно станет незаменимым помощником для всех, кто стремится систематизировать свои тренировки и добиться желаемых результатов.

2.4 Тестирование

При разработке приложения проводилось ручное тестирование, в ходе которого было выявлено и исправлено множество багов. Одним из обнаруженных багов связан с удалением упражнения, когда его подходы оставались в системе. Баг продемонстрирован на рисунке 26.

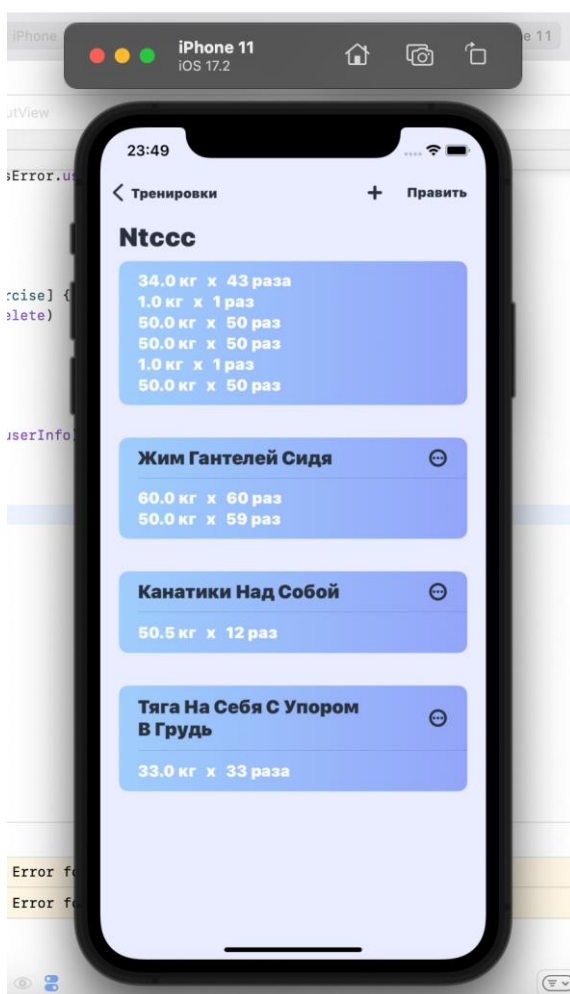


Рисунок 26. Сохранение подходов после удаления упражнения

Взаимодействие с этими подходами приводило к вылету приложения. Также выявлена особенность, связанная с различиями европейской и американской раскладками клавиатур, в частности с отличием символов "," и ".". Различие отображено на рисунке 27.



Рисунок 27. Региональное различие клавиатур

Введение дробных значений веса пользователями европейских стран приводило к вылету приложения, так как работа кода завязана на точке, а не запятой. Для решения проблемы был реализован код позволяющий заменить запятую точкой. Код представлен на рисунке 28.

```
TextField("", text: Binding(
  get: { self.newWeight },
  set: { self.newWeight = $0.replacingOccurrences(of: ",", with: ".") }
), prompt: Text("Вес").foregroundColor(.white))
.textFieldStyle()
.font(.system(size: 20))
.frame(height: 50)
.keyboardType(.decimalPad)
```

Рисунок 28. Код для замены запятой

get возвращает текущее значение переменной newWeight, а set устанавливает новое значение для переменной newWeight. Введённая запятая будет автоматически заменена на точку.

Кроме того, ввод отрицательных чисел или текста приводил к вылету приложения. Было написано условие для обработки некорректных значений.

После создания персонального стиля для навигационных кнопок возник баг с исчезновением кнопки "назад". Решением данной проблемы стало возвращение размера шрифта к стандартному значению.

На этапе тестирования было выявлено, что добавление нового упражнения через настройки занимает много времени, поэтому был реализован дополнительный функционал. Пользователь может выбрать упражнение из списка предустановленных или создать упражнение со своим названием без необходимости отправляться в настройки и добавлять упражнение в список упражнений.

Ручное тестирование позволило выявить и оперативно устранить множество багов, значительно улучшив стабильность и удобство использования приложения.

2.5 Перспективы

Будущее развитие приложения «Gym Tracker» будет направлено на расширение функциональности и улучшение пользовательского опыта. Одной из

новых функций станет система создания собственных категорий, что даст пользователям больше возможностей для персонализации приложения.

Планируется внедрение функции изменения уже созданных подходов, что значительно повысит удобство использования.

Введение функции отображения информации о предыдущей тренировке и выполненных упражнениях позволит пользователям сравнивать текущие результаты с прошлыми. Это нововведение ускорит процесс заполнения подходов, так как пропадёт надобность в открытии экрана тренировок. Пользователи смогут легко и быстро корректировать свой тренировочный план.

Визуализация статистики с помощью графиков предоставит пользователям наглядное представление об их прогрессе. Графики будут отображать изменения в весе, количестве повторений и других ключевых метриках, что позволит пользователям лучше понимать свои достижения.

Кроме того, планируется реализация функции заготовленных тренировок, которая позволит пользователям быстро начинать тренировки по заранее подготовленным программам.

Локализация на популярные языки станет важным шагом для расширения потенциальной аудитории.

В конечном счёте приложение будет размещено в App Store, что обеспечит доступ к огромной аудитории пользователей iOS-устройств. Обширный набор функций повысит конкурентоспособность, позволит удовлетворять потребности различных категорий пользователей и занять лидирующие позиции на рынке.

ЗАКЛЮЧЕНИЕ

При выполнении выпускной квалификационной работы было разработано мобильное приложение для учёта тренировок «Gym Tracker».

По мере достижения цели работы были решены следующие задачи:

- На основе проведённого анализа выявлены преимущества и недостатки существующих приложений для учёта тренировок;
- На основе проведённого анализа была выбрана подходящая технология для реализации приложения;
- Описаны требования для разрабатываемого приложения;
- Создан прототип мобильного приложения, согласно описанным требованиям;
- Описан процесс разработки и тестирования приложения.

Все задачи, поставленные в выпускной квалификационной работе, полностью решены, а цель достигнута.

СПИСОК ЛИТЕРАТУРЫ