**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»**

(ВлГУ)

Кафедра ВТиСУ

КУРСОВАЯ РАБОТА

по дисциплине «Разработка мобильных приложений»

на тему: «Разработка приложения для контроля тренировок и питания»

Студент: Истратов И.С.

Институт: ИИТЭ

Направление: 09.03.03

Группа: ПИ-121

Руководитель: Шутов А. В.

Владимир 2024 г.

**Оглавление**

[Введение 3](#_Toc188521334)

[1. Теоретическая часть 4](#_Toc188521335)

[1.1 Понятие и виды мобильных приложений 4](#_Toc188521336)

[1.2 Описание предметной области и задач на разработку 5](#_Toc188521337)

[2. Практическая часть 7](#_Toc188521338)

[2.1 Анализ и выбор СMS системы для разработки серверной части мобильного приложения 7](#_Toc188521339)

[2.2 Обзор платформ для разработки мобильного приложения 9](#_Toc188521340)

[2.3 Архитектура приложения 9](#_Toc188521341)

[2.4 Обзор разработанного приложения 17](#_Toc188521342)

[Заключение 22](#_Toc188521343)

[Список литературы 23](#_Toc188521344)

# **Введение**

Современный мир активно движется к здоровому образу жизни, а цифровые технологии играют важную роль в этом процессе. Мобильные приложения стали незаменимым инструментом для тех, кто стремится улучшить свое физическое состояние, предлагая удобный доступ к тренировкам и планам питания в любое время и из любого места. Одной из наиболее востребованных областей является сфера фитнеса и диетологии, где использование мобильных технологий позволяет персонализировать программы тренировок и питания, а также повысить мотивацию пользователей.

Целью данной работы является разработка мобильного приложения, предоставляющего пользователям интуитивно понятный интерфейс и функционал для эффективного планирования тренировок и контроля рациона питания.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить инструменты и технологии, применимые при разработке приложений на платформе Android.
2. Разработать удобный и понятный интерфейс, соответствующий потребностям пользователей.
3. Реализовать основные функции приложения, включая составление программ тренировок и рекомендаций по питанию.
4. Внедрить возможность отслеживания прогресса и управления персональными данными.

Таким образом, данная курсовая работа направлена на приобретение теоретических знаний и практических навыков в области разработки мобильных приложений для Android, что соответствует требованиям образовательной программы и способствует профессиональному развитию в сфере информационных технологий.

# **1. Теоретическая часть**

## **1.1 Понятие и виды мобильных приложений**

Мобильные приложения — это программное обеспечение, предназначенное для работы на смартфонах, планшетах и других портативных устройствах. Они помогают пользователям решать разнообразные задачи: от общения и получения информации до развлечений и работы. Для установки мобильных приложений используются официальные платформы, такие как App Store (для iOS) и Google Play (для Android).

**Виды мобильных приложений**

1. Нативные приложения — разрабатываются специально для одной операционной системы (iOS или Android) и используют соответствующие языки программирования (Swift, Objective-C для iOS; Java, Kotlin для Android). Такие приложения обеспечивают максимальную производительность и полный доступ к функциям устройства.
2. Гибридные приложения — сочетают элементы нативных и веб-приложений, создаются с использованием HTML, CSS, JavaScript и работают внутри оболочки нативного интерфейса. Это позволяет запускать одно приложение сразу на нескольких платформах.
3. Веб-приложения — представляют собой адаптированные веб-сайты, которые работают через браузер без необходимости установки. Обновления происходят на сервере, что гарантирует пользователям доступ к актуальной версии приложения.
4. Прогрессивные веб-приложения (PWA) — это улучшенные веб-приложения, обладающие возможностями нативных: они могут работать в оффлайн-режиме, отправлять уведомления и устанавливаться на устройство.
5. Игровые приложения — создаются для развлечения и взаимодействия с пользователем через игровую механику. Могут быть как нативными, так и гибридными, зачастую требуют значительных ресурсов для обработки графики.

**Основные характеристики мобильных приложений**

Доступность – удобная загрузка и установка через официальные магазины.

Интерактивность – приложения обеспечивают удобный интерфейс и быстрый отклик на действия пользователя.

Интернет-соединение – многие программы требуют подключения к сети для обновления контента и работы основных функций.

Мобильные приложения продолжают играть важную роль в повседневной жизни, обеспечивая пользователям удобные инструменты для работы, учебы, общения и развлечений.

## **1.2 Описание предметной области и задач на разработку**

Предметной областью разработки мобильного приложения является сфера фитнеса и здорового питания. Основной целью является создание удобного и интуитивно понятного мобильного приложения, которое позволит пользователям эффективно планировать тренировки и составлять рацион питания. Приложение предоставит доступ к персонализированным программам упражнений, рекомендациям по питанию, а также возможностям отслеживания прогресса.

Функциональность для пользователей:

* Выбор и настройка программ тренировок в зависимости от уровня подготовки и целей.
* Доступ к базам данных упражнений с описаниями и видеоинструкциями.
* Составление индивидуального плана питания с учетом калорийности и БЖУ.
* Отслеживание прогресса, включая вес, объемы тела и достижения в тренировках.

Функциональность для администратора:

* Управление контентом (добавление, редактирование и удаление тренировочных программ и рекомендаций по питанию).
* Анализ активности пользователей для улучшения рекомендаций.

Основные пользователи приложения:

* Клиенты, желающие организовать свой режим тренировок и питания.
* Администратор, отвечающий за обновление контента и анализ данных.

Ключевые задачи проекта:

1. Разработка мобильного приложения для платформы Android с интуитивно понятным интерфейсом.
2. Реализация возможностей поиска и фильтрации программ тренировок и планов питания.
3. Создание интерактивных карточек упражнений и рецептов с подробным описанием и фото/видео.
4. Внедрение системы отслеживания прогресса и рекомендаций по улучшению результатов.

Данное приложение поможет пользователям выстроить здоровый образ жизни, контролировать физическую активность и питание, а также достигать поставленных целей в сфере фитнеса.

# **2. Практическая часть**

## **2.1 Анализ и выбор СMS системы для разработки серверной части мобильного приложения**

**Firebase** – облачная платформа от Google для мобильных и веб-приложений

Firebase предоставляет набор облачных сервисов, позволяющих разработчикам легко управлять базами данных, аутентификацией, хостингом и аналитикой.

Преимущества:

* Готовые решения для бэкенда – база данных Firestore, хостинг, аутентификация, push-уведомления.
* Реактивные базы данных – изменения в данных моментально отображаются в приложении.
* Интеграция с Google Cloud – расширенные возможности для масштабирования.
* Мощная аналитика – встроенная поддержка Google Analytics.

Недостатки:

* Закрытая экосистема – жесткая привязка к Google Cloud.
* Ограничения бесплатного тарифа – при росте нагрузки могут быть значительные затраты.
* Неструктурированные базы данных – для сложных реляционных данных могут возникнуть трудности.

**Supabase** – альтернатива Firebase с открытым исходным кодом

Supabase – это open-source решение, построенное на базе PostgreSQL, предоставляющее возможности реального времени и удобный API.

Преимущества:

* Использует SQL-базу данных – мощный и гибкий PostgreSQL.
* Простая настройка API – автоматически создается REST и GraphQL API.
* Легкость миграции – совместим с традиционными SQL-решениями.
* Открытый исходный код – можно развернуть на своем сервере.

Недостатки:

* Меньше встроенных инструментов по сравнению с Firebase.
* Требует дополнительных настроек для масштабирования.
* Ограниченные серверные функции в сравнении с Google Cloud.

**Strapi** – это open-source headless CMS, которая позволяет разработчикам быстро создать API для управления контентом. Она предлагает средства для создания пользовательских типов контента и имеет встроенную поддержку GraphQL.

Преимущества:

* Полный контроль над кодом и возможностью хостинга на собственных серверах.
* Позволяет легко создавать REST и GraphQL API, по сравнению с другими CMS.
* Удобная панель управления для администраторов и контент-менеджеров.
* Поддержка плагинов и расширений для кастомизации.

Недостатки:

* В больших проектах может требовать оптимизации.
* Требует дополнительных усилий для настройки и развертывания.
* Меньшая активность сообщества по сравнению с более крупными решениями.

Каждая из этих платформ предлагает уникальные инструменты и подходы к управлению контентом, и выбор между ними зависит от потребностей конкретного проекта и команды.

Для разработки мобильного приложения было принято решение использовать Strapi, по следующим причинам:

1. Strapi — это headless CMS, что означает, что он позволяет вам создавать и управлять контентом без жесткой привязки к конкретному фронтенду. Можно использовать любой фреймворк или библиотеку (React, Vue, Angular и т.д.), что дает большую гибкость в разработке.

2. Strapi автоматически генерирует RESTful и GraphQL API, что позволяет легко интегрировать приложение с другими сервисами и использовать современные подходы к разработке.

3. Strapi поддерживает плагины и механизмы кастомизации, что позволяет разработчикам добавлять новые функции и интегрировать сторонние сервисы согласно специфическим требованиям проекта.

4. Strapi позволяет работать с несколькими базами данных, такими как MongoDB, PostgreSQL, MySQL и SQLite, что позволяет выбрать оптимальное решение для вашего приложения.

5. Strapi предлагает встроенные механизмы управления пользователями и их правами доступа, что обеспечивает защиту данных на уровне API и управления контентом.

6. Strapi позволяет разрабатывать приложения как для веба, так и для мобильных устройств, что делает Strapi идеальным решением для многоплатформенной разработки.

В то время как Firebase и Supabase предлагают свои преимущества, такие как облачные решения и интеграции, Strapi выделяется своей гибкостью, открытостью и возможностью самохостинга, что делает его привлекательным выбором для разработчиков, ищущих контроль и настройку под свои нужды.

**2.2 Обзор платформ для разработки мобильного приложения**

Для разработки приложения на Android существует огромное множество платформ, рассмотрим некоторые из них:

**NativeScript** – нативные мобильные приложения на JavaScript/TypeScript

NativeScript – это фреймворк для разработки кроссплатформенных мобильных приложений с использованием JavaScript, TypeScript или Angular. Он позволяет использовать нативные API Android и iOS, сохраняя при этом удобство фронтенд-разработки.

Ключевые особенности:

* Прямой доступ к нативным API без необходимости писать обертки.
* Поддержка популярных фреймворков – можно использовать Angular, Vue или чистый TypeScript.
* Высокая производительность, так как код напрямую взаимодействует с нативными компонентами.
* Открытый исходный код, что позволяет кастомизировать платформу под свои нужды.

Недостатки:

* Меньшее комьюнити, чем у React Native и Kotlin Multiplatform.
* Размер приложений может быть больше, чем у других решений, из-за включенных зависимостей.

**Kotlin Multiplatform (KMP)** – гибкое решение для Android и iOS.

Kotlin Multiplatform – это технология от JetBrains, которая позволяет писать общий код на Kotlin для Android, iOS, Web и Desktop, при этом давая возможность использовать нативные API там, где это необходимо.

Ключевые особенности:

* Использование общего кода для бизнес-логики, с возможностью написания платформенно-специфичных модулей.
* Гибкость – позволяет использовать нативные инструменты UI для каждой платформы.
* Хорошая интеграция с Android – Kotlin является официальным языком разработки для платформы.
* Производительность близка к нативной, так как код компилируется отдельно для каждой платформы.

Недостатки:

* Требуется писать нативный UI-код, что может увеличить время разработки.
* Меньше готовых решений, чем у React Native, что может потребовать больше работы с интеграцией сторонних библиотек.

**React Native** – мощное кроссплатформенное решение от Meta

React Native – это фреймворк с открытым исходным кодом, который позволяет создавать мобильные приложения для Android и iOS на JavaScript и React. Благодаря активному сообществу и большому количеству готовых библиотек, разработка на React Native отличается высокой скоростью и удобством.

Ключевые особенности:

* Единая кодовая база для Android и iOS, что снижает время и затраты на разработку.
* Горячая перезагрузка позволяет мгновенно видеть изменения в коде без пересборки.
* Богатая экосистема и интеграция с популярными библиотеками React.
* Поддержка нативных компонентов, обеспечивающая высокую производительность.

Недостатки:

* В ряде случаев требуется написание нативного кода для сложных функций.
* Производительность может уступать некоторым нативным решениям при сложных графических задачах.

## **Выбор платформы для мобильного приложения**

Для разработки мобильного приложения, ориентированного на тренировки и питание, был выбран React Native, поскольку он соответствует ключевым требованиям проекта:

* Быстрота разработки – благодаря использованию JavaScript и множества готовых библиотек можно оперативно внедрять новые функции и адаптировать приложение под пользовательские запросы.
* Кроссплатформенность – единый код позволяет разрабатывать и поддерживать приложение одновременно для Android и iOS, сокращая временные и финансовые затраты.
* Активное сообщество и поддержка – наличие большого числа готовых решений и документации облегчает разработку и ускоряет исправление возможных ошибок.
* Гибкость интеграции – React Native без проблем взаимодействует с серверной частью на Strapi, что упрощает управление контентом и обработку данных пользователей.
* Оптимизированный пользовательский интерфейс – поддержка нативных компонентов помогает создавать удобный и отзывчивый интерфейс, что особенно важно для фитнес-приложений.

Несмотря на то, что другие решения, такие как Kotlin Multiplatform, обеспечивают высокий уровень производительности и глубокой интеграции с нативными API, React Native был выбран за счет удобства работы, скорости внедрения новых функций и широких возможностей кастомизации.

## **2.3 Архитектура приложения.**

Архитектура мобильного приложения для планирования тренировок и питания основана на современных технологиях, обеспечивающих производительность, удобство разработки и масштабируемость. В основе лежит клиент-серверная модель, где серверная часть реализована на Node.js с CMS Strapi, а мобильное приложение взаимодействует с сервером через REST API.

**Клиентская сторона (React Native)**

Приложение на React Native предоставляет интуитивно понятный интерфейс, упрощая взаимодействие пользователей с системой. Архитектура состоит из нескольких ключевых компонентов:

1. Пользовательский интерфейс (UI)

* Включает экраны для отображения расписания тренировок, плана питания, статистики прогресса.
* Обеспечивает удобную навигацию и интуитивное взаимодействие.

1. Модель (Model)

* Управляет данными о тренировках, питании, достижениях пользователя.
* Синхронизируется с сервером для получения актуальной информации.

1. Контроллер (Controller)

* Обрабатывает пользовательские действия, например, выбор программы тренировок или добавление продуктов в дневник питания.
* Реализует бизнес-логику взаимодействия между UI и моделью.

1. Сетевой слой

* Отвечает за асинхронное взаимодействие с сервером, обеспечивая бесперебойную работу.
* Использует Axios для выполнения HTTP-запросов (получение данных, отправка пользовательских параметров и результатов тренировок).

**Серверная сторона (Node.js с CMS Strapi):**

1. **CMS Strapi:**

* Управляет контентом, включая программы тренировок, советы по питанию и статистику.
* Автоматически генерирует API для взаимодействия с клиентом.

1. **REST API:**

* Поддерживает CRUD-операции:
* GET – получение списка тренировок или данных о пользователе.
* POST – добавление новой тренировки или сохранение дневника питания.
* PUT – обновление параметров пользователя.
* DELETE – удаление записей о тренировках.

**3.** **База данных (SQLite):**

* Хранит данные пользователей, их тренировки, пищевые дневники и статистику прогресса.
* Поддерживает масштабируемость и обеспечивает быструю обработку запросов.

Взаимодействие компонентов:

1. UI

* Пользователь выбирает программу тренировок, добавляет приемы пищи или анализирует прогресс.
* Приложение отправляет HTTP-запрос на сервер.

1. HTTP-запрос

* Отправляется с помощью Axios.

Запросы бывают:

* GET – получение списка упражнений, блюд, статистики.
* POST – сохранение новой тренировки или рациона.

1. Обработка сервером

* Strapi обрабатывает запрос, взаимодействует с PostgreSQL и выполняет операции по выборке, обновлению или удалению данных.

1. База данных (SQLite)

* Хранит и обрабатывает данные пользователей, их план тренировок, калории и физические показатели.

1. Ответ клиенту

* Сервер отправляет ответ в формате JSON (например, обновленные данные о калориях и прогрессе).

1. Обновление интерфейса

* Полученный ответ обновляет UI, чтобы пользователь видел актуальную информацию.

**Преимущества архитектуры:**

1. Модульность и масштабируемость:

* Использование CMS Strapi упрощает управление контентом и API.

2. Удобство разработки:

* React Native и SQLite обеспечивают лёгкость интеграции и высокую производительность.

3. Гибкость:

* REST API позволяет адаптировать функционал под новые требования.

4. Кроссплатформенность:

* React Native позволяет использовать приложение как на Android, так и на iOS.

База данных не имеет графической оболочки.

Эта архитектура гарантирует **высокую производительность, удобство использования** и возможность **дальнейшего развития** приложения для занятий спортом и контроля питания. туристического агентства.

## **2.4 Обзор разработанного приложения**

При открытии приложения пользователю отображается экран регистрации с логином и паролем (рис. 1).

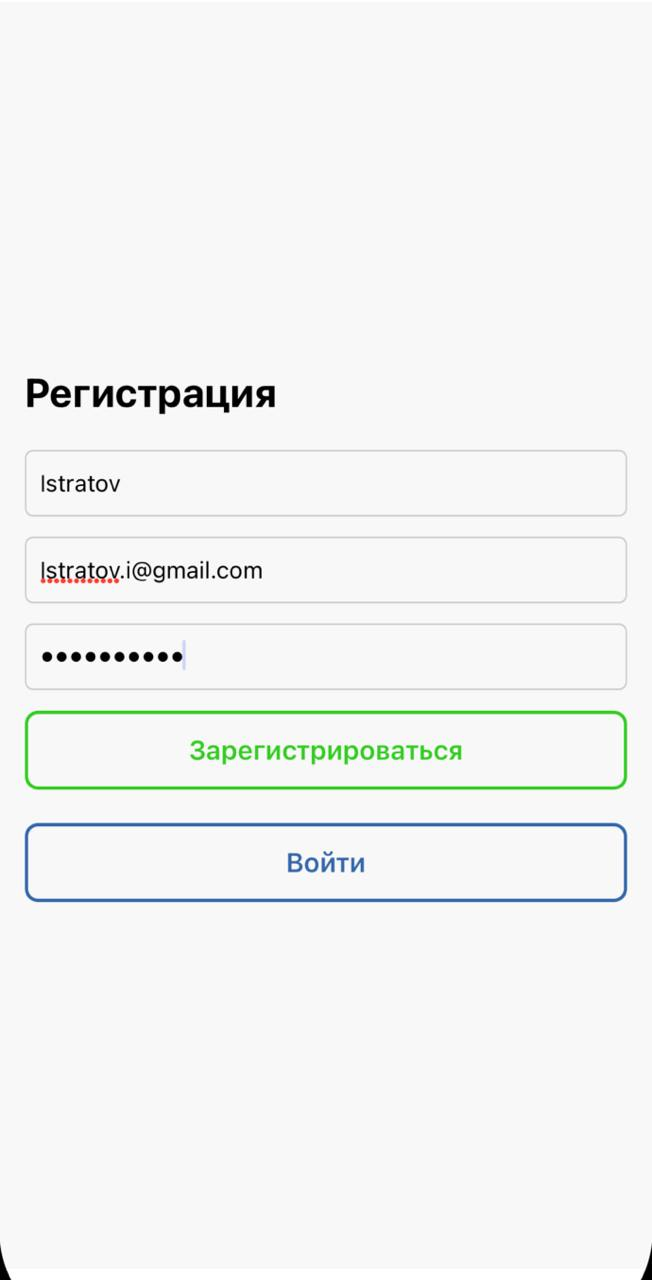


Рисунок 1 – Регистрация.

После регистрации в приложении пользователю отображается главный экран с профилем пользователя и его показателями (рис. 2-3). Основной экран содержит следующие элементы:

1. Профиль пользователя.

2. График веса пользователя, калорийности питания.

3. Нижняя панель навигации.

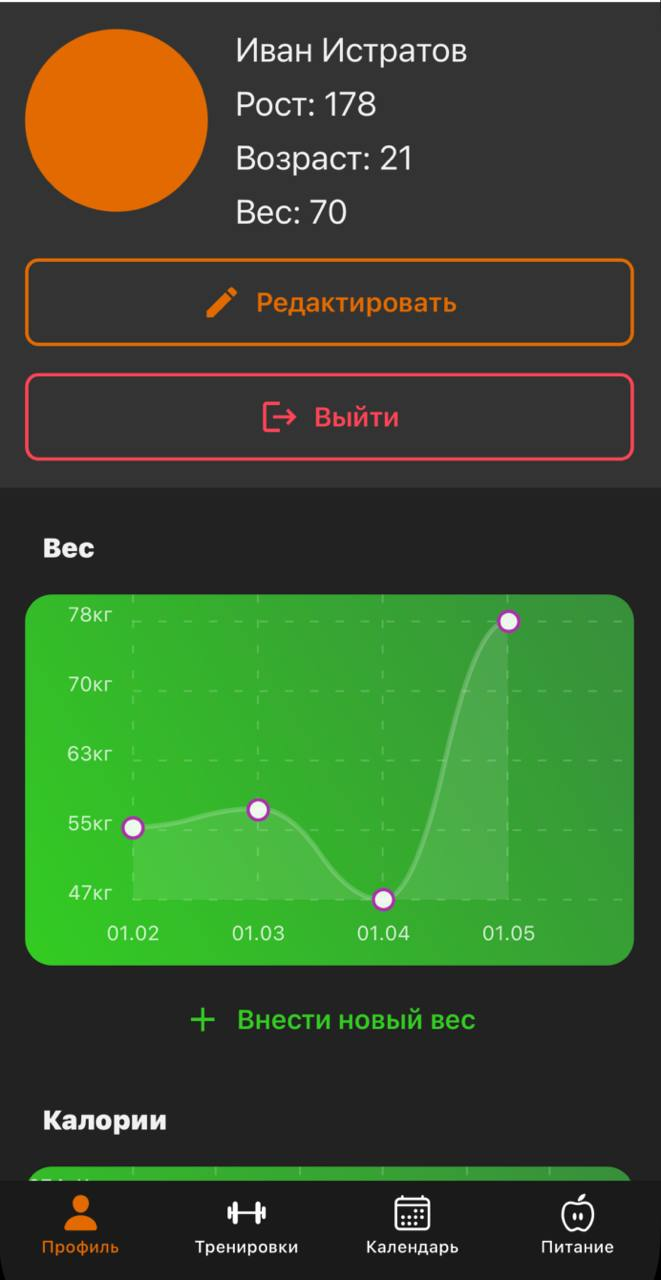
 

Рисунок 2, 3 – Главная страница.

На экране «Тренировки» пользователь может добавить новую тренировку (в том числе упражнение) и увидеть проведенные в этот день тренировки (рис. 4-5).

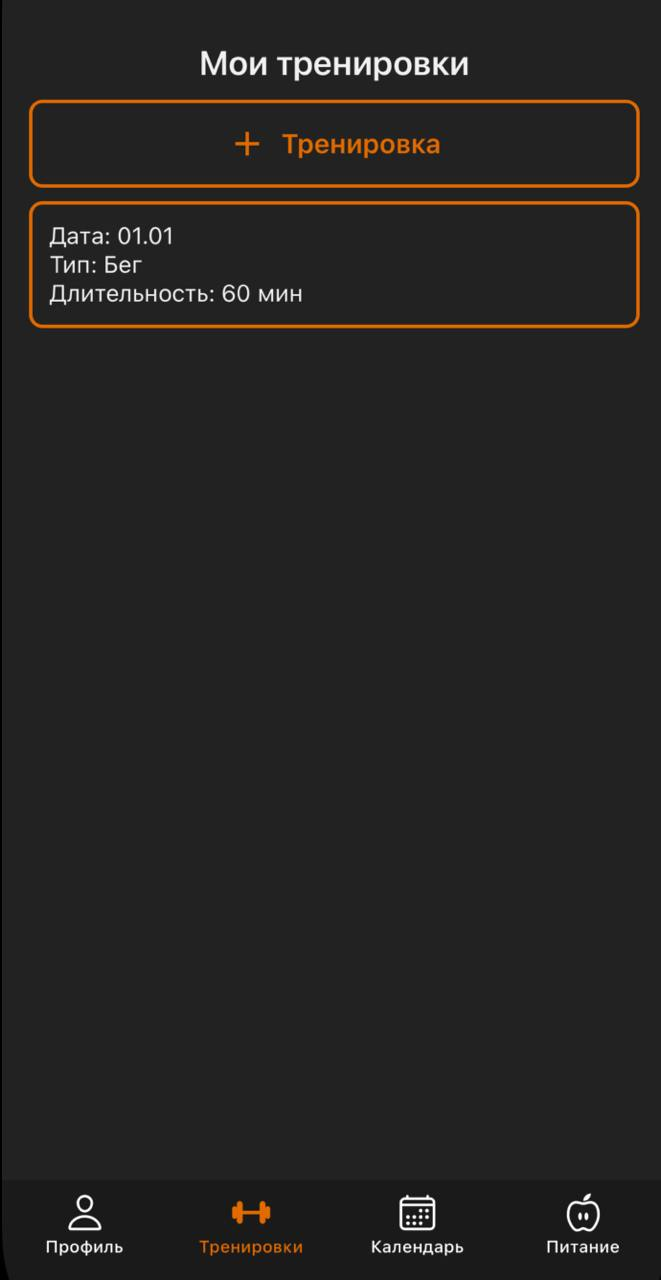
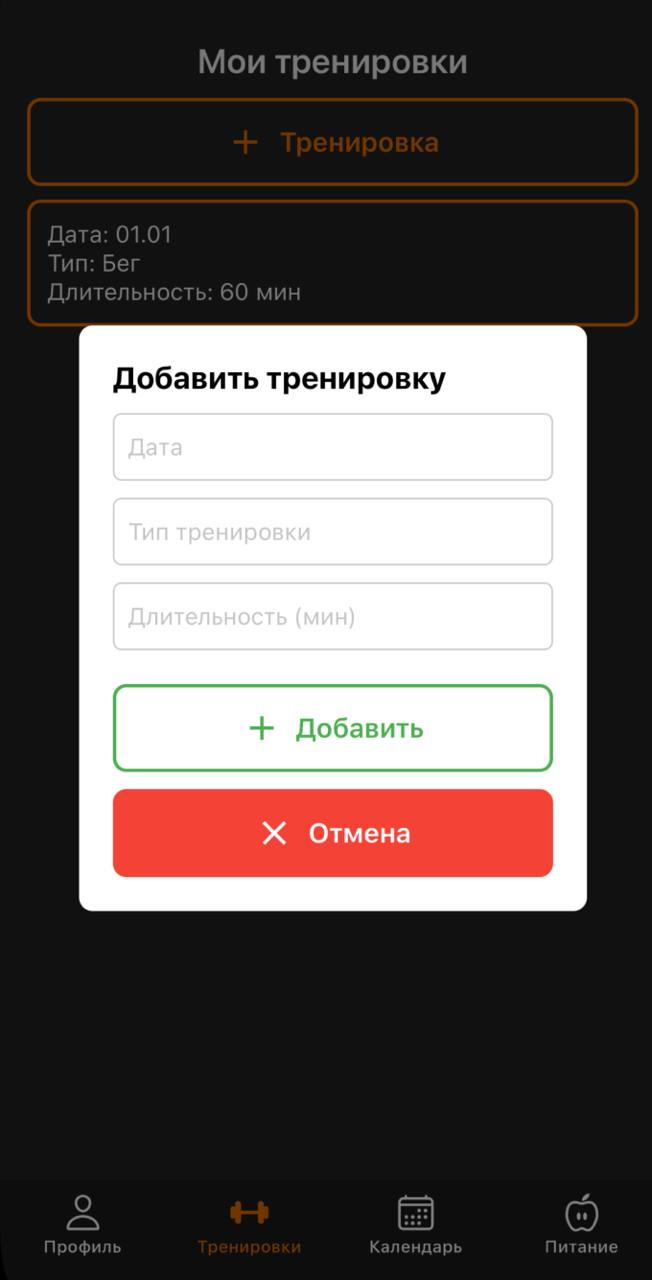
 

Рисунок 4-5 – Раздел «Тренировки».

На странице «Календарь» пользователь может отслеживать тренировки и приёмы пищи за выбранный период (рис. 6)

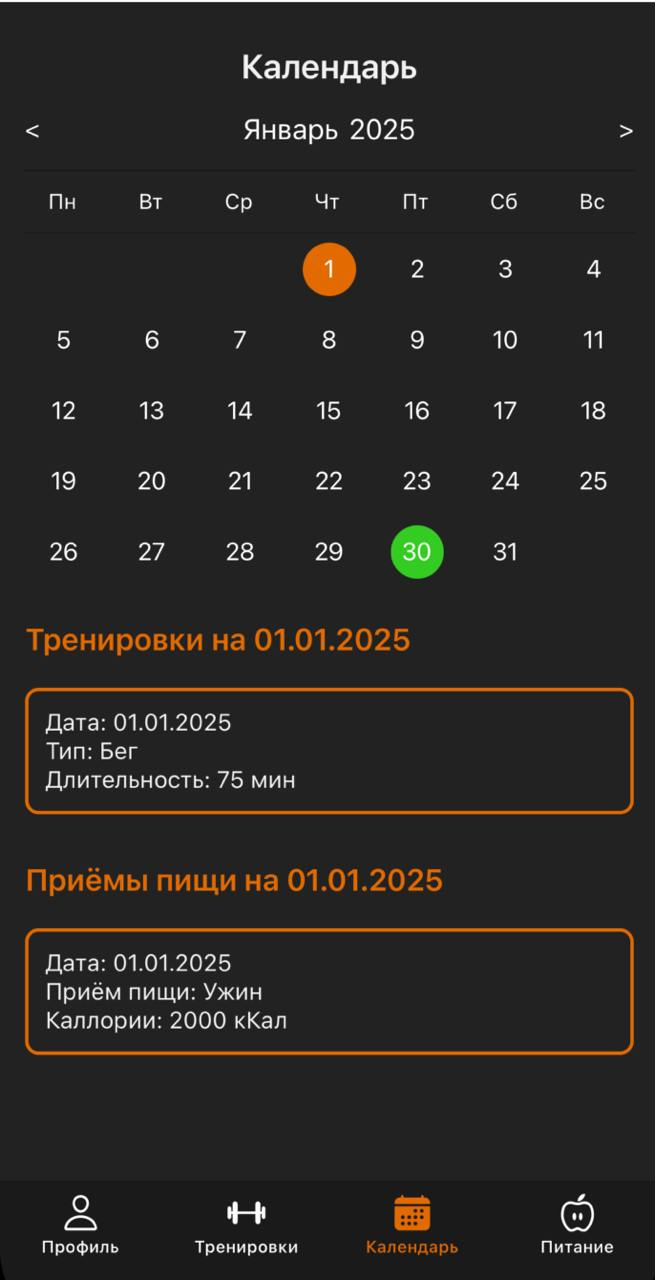


Рисунок 6 – Экран «Календарь».

На странице «Питание» пользователь может добавить приём пищи и её калорийность (рис. 7)

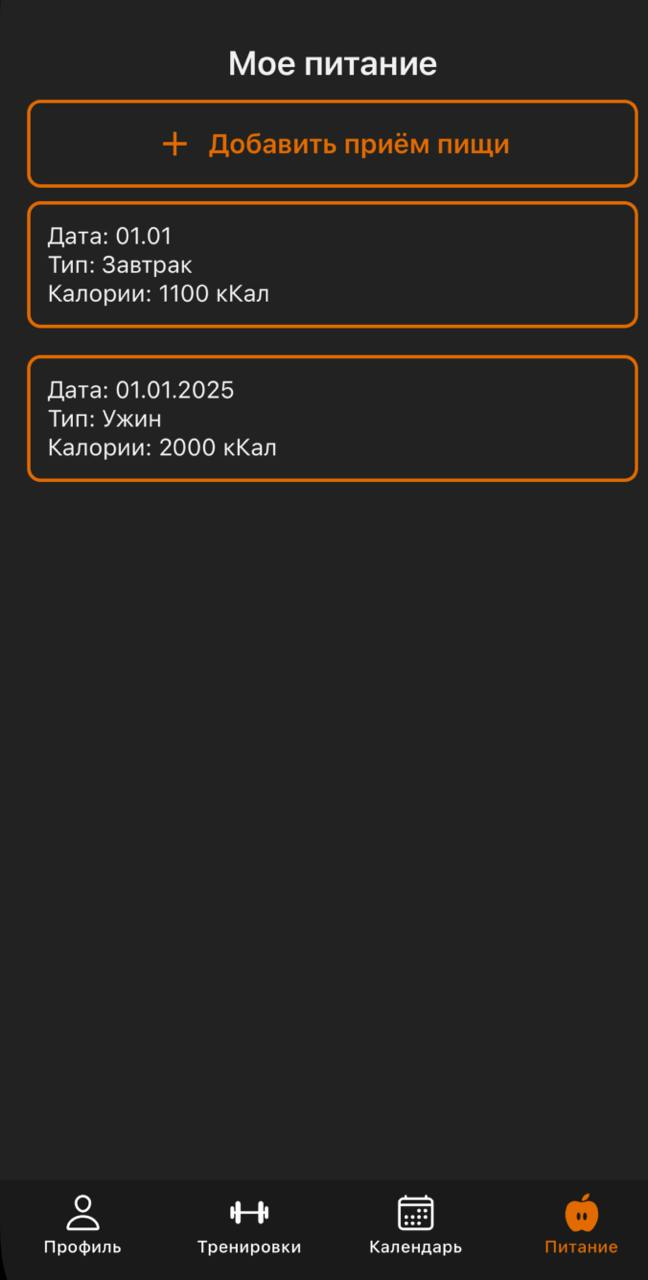


Рисунок 7 – Страница «Питание».

# **Заключение**

В процессе разработки мобильного приложения для планирования тренировок и питания был выполнен полный цикл работ: от анализа требований и постановки задач до создания полнофункционального программного продукта. Приложение включает все ключевые возможности, необходимые для удобного составления программ тренировок и рационов питания.

Достигнутые результаты:

* Проведено исследование современных трендов в сфере фитнеса и здорового питания, что позволило выделить ключевые требования к функционалу.
* Учет актуальных тенденций способствовал созданию удобного и востребованного продукта.
* Разработаны интерфейсы для составления персонализированных программ тренировок и планов питания.
* Добавлен трекер прогресса для повышения мотивации пользователей.
* Пользователи могут настраивать планы тренировок и питания в зависимости от своих целей.
* Приложение способствует формированию здоровых привычек, улучшению физической формы и повышению уровня самодисциплины.
* Упрощает процесс планирования тренировок и контроля питания, повышая удобство для пользователей.

# **Список литературы**

1. Мозес, Э. React Native в действии. — М.: ДМК Пресс, 2022. — 320 с.
2. Стойлов, А. В. Разработка мобильных приложений на React Native. — СПб.: Питер, 2021. — 288 с.
3. Саймон, Р., Браун, Ч. SQLite для разработчиков. — М.: БХВ-Петербург, 2020. — 240 с.
4. Миллер, Д. API и управление контентом с помощью Strapi. — М.: Альпина Паблишер, 2022. — 256 с.