МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет Компьютерных наук

Кафедра программирования и информационных технологий

Мобильный фитнес-тренер: приложение для домашних тренировок. «Trainova»

Курсовая работа

Направление: 09.03.02 Информационные системы и технологии

Зав. Кафедрой _____ д. ф.-м. н, доцент С.Д. Махортов

Руководитель _____ ст. преподаватель В.С. Тарасов

Руководитель практики _____ ассистент М.А.Савин

Обучающийся _____ Е.Ю. Мануковский, 3 курс, д/о

Обучающийся _____ А.Е. Анисимова, 3 курс, д/о

Обучающийся _____ Д.А. Колчин, 3 курс, д/о

Обучающийся _____ А.К. Быханов, 3 курс, д/о

Обучающийся _____ К.В. Курманов, 3 курс, д/о

Обучающийся _____ Д.В. Косьянов, 3 курс, д/о

Воронеж 2025

Содержание

Термины и определения

— **Блок А**– условное обозначение, придуманное командой разработки для обозначения блока приложения с тренировками от приложения;

— **Блок Б**– условное обозначение, придуманное командой разработки для обозначения блока приложения с тренировками от тренеров;

— **Курс**– сборник тренировок, составленный тренером и доступный бесплатно или по подписке;

— **Backend**– серверная часть продукта, которая отвечает за внутреннюю логику и работу мобильного приложения;

— **Frontend**– Пользовательский интерфейс, к нему относится всё, что пользователи видят в мобильном приложении, и с чем можно взаимодействовать;

— **Администратор**– авторизованный пользователь с особыми привилегиями, человек на эту роль назначается заказчиком;

— **API** – это программный интерфейс приложений, набор инструкций, который позволяет разным приложениям общаться между собой[1].

— **Docker**– платформа с открытым исходным кодом для автоматизации разработки, доставки и развёртывания приложений;

— **React**– JavaScript-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов;

— **Next.js**– фреймворк на базе React для разработки клиент-серверных (fullstack) приложений как для мобильных платформ, так и для веба;

— **Nginx**– программное обеспечение с открытым исходным кодом для создания веб-серверов;

— **GitHub**– веб-сервис для хостинга IT-проектов и их совместной разработки;

— **PWA (Progressive Web Apps)**– тип веб-приложения, который сочетает в себе преимущества веб-сайтов и нативных мобильных приложений.

Введение

Данная курсовая работа посвящена разработке мобильного приложения «Trainova» – платформы, предназначенной для организации домашних тренировок. Приложение разработано для пользователей с разным уровнем физической подготовки, предоставляя как новичкам, так и опытным спортсменам доступ к персонализированным тренировочным программам, включая бесплатные тренировки от платформы и платные курсы от профессиональных тренеров.

В последние годы мобильные приложения для фитнеса приобретают все большую популярность благодаря их удобству и способности удовлетворять потребности пользователей, предпочитающих тренироваться дома. Рост интереса к здоровому образу жизни и доступность цифровых технологий подчеркивают актуальность создания таких решений. «Trainova» выделяется среди аналогов благодаря функционалу, позволяющему фильтровать тренировки по группам мышц, оценивать и комментировать курсы, а также получать мотивационные сообщения, созданные с использованием модели искусственного интеллекта DeepSeek R1. Важной особенностью является возможность для тренеров выкладывать свои тренировочные курсы через специальный конструктор, что делает платформу привлекательной для профессионалов, стремящихся расширить свою аудиторию.

Целью работы является создание приложения, которое позволит пользователям заниматься спортом дома. Внимание уделяется созданию понятного пользовательского интерфейса, обеспечению безопасности данных и поддержке масштабируемости платформы. В рамках курсовой работы будут проанализированы потребности целевой аудитории, потенциальные технические риски и предложены подходы к их минимизации, чтобы заложить основу для создания конкурентоспособного продукта на рынке фитнес-приложений.

1 Проблемы исследования

Разработка мобильного приложения «Trainova» для персонализированных домашних тренировок сталкивается с рядом значимых проблем, которые определяют актуальность и сложность проекта. На основе анализа представленных данных и визуальных материалов выделяются следующие ключевые аспекты проблематики:

1.Составление тренировок. Процесс создания эффективных и безопасных тренировочных программ требует учета индивидуальных особенностей пользователей, таких как уровень подготовки и цели. Отсутствие автоматизированных инструментов для адаптации тренировок может привести к снижению их эффективности и увеличению риска травм.

2.Сложность самоконтроля. Пользователи часто сталкиваются с трудностями в отслеживании своего прогресса и соблюдении режима тренировок без профессионального надзора. Это требует разработки интуитивно понятных инструментов мониторинга и мотивации, что представляет техническую и психологическую задачу.

3. Поиск аудитории. Привлечение целевой аудитории, заинтересованной в домашних тренировках, осложняется высокой конкуренцией и необходимостью продвижения уникальных возможностей приложения. Необходимость четкого позиционирования и маркетинговой стратегии добавляет сложность проекту.

Таким образом, проблемы заключаются в необходимости разработки решения, которое обеспечит безопасные тренировки, позволит пользователям отслеживать свой прогресс и обеспечит привлечение аудитории для тренеров. Успешное решение этих задач позволит создать конкурентоспособное приложение, удовлетворяющее потребности пользователей.

2 Назначение и цель создания приложения

2.1 Цели создания приложения

Получение 20% от ежемесячных подписок пользователей на платные курсы тренировок.

2.2 Задачи, решаемые с помощью приложения

Разрабатываемое приложение решает следующие задачи:

- предоставление доступа к персонализированным тренировочным программам для пользователей с разным уровнем физической подготовки, что позволяет эффективно и безопасно заниматься спортом дома.
- упрощение самоконтроля и мотивации пользователей за счет предоставления инструментов для отслеживания прогресса (графиков веса и активности) и генерации мотивационных сообщений с использованием искусственного интеллекта.
- создание платформы для тренеров, позволяющей публиковать и монетизировать собственные обучающие курсы, а также привлекать новую аудиторию.
- обеспечение взаимодействия между пользователями и тренерами посредством просмотра, оценки и комментирования курсов, что способствует повышению качества контента и удовлетворенности пользователей.

3 Целевая аудитория

Продукт ориентирован на пользователей, которые интересуются спортом и предпочитают заниматься дома. Основные характеристики целевой аудитории:

- наличие интереса к спортивным занятиям и здоровому образу жизни.

— желание тренироваться в домашних условиях подчеркивает потребность в удобных и доступных решениях для самостоятельных тренировок.

— для тренеров – это возможность публиковать свои курсы, монетизировать их и привлекать новых клиентов через платформу

Приложение «Trainova» разработано с учетом этих потребностей и предоставляет пользователям доступ к персонализированным тренировочным программам, инструментам для отслеживания прогресса и мотивации, а также возможность выбора курсов от профессиональных тренеров. Таким образом, продукт отвечает запросам аудитории, стремящейся к эффективным и комфортным домашним тренировкам.

4 Обзор конкурентов

Среди конкурентов фитнес-приложений были рассмотрены крупнейшие и наиболее популярные платформы, предлагающие онлайн-тренировки: «Nike Training Club», «Adidas Training», «Fitify», «Freeletics» и «FitOn».

В отличие от «Trainova», большинство этих приложений ориентированы либо на самостоятельные тренировки без обратной связи с тренером, либо на готовые универсальные программы. Приложение «Trainova» ориентировано на гибридную модель, в которой пользователь может как самостоятельно заниматься по общим материалам, так и приобрести видеокурсы, созданные профессиональными тренерами, получив доступ к их тренировочным программам. Таким образом, существующие приложения являются косвенными конкурентами, не охватывающими весь функционал «Trainova».

Ниже приведён краткий анализ ключевых конкурентов:

4.1 Nike Training Club

Nike Training Club – бесплатное мобильное приложение, разработанное для поддержания здоровья и физической формы пользователей. В нём

собраны различные тренировки, разработанные профессиональными тренерами.

Вот основные преимущества и недостатки Nike Training Club по сравнению с нашим приложением.

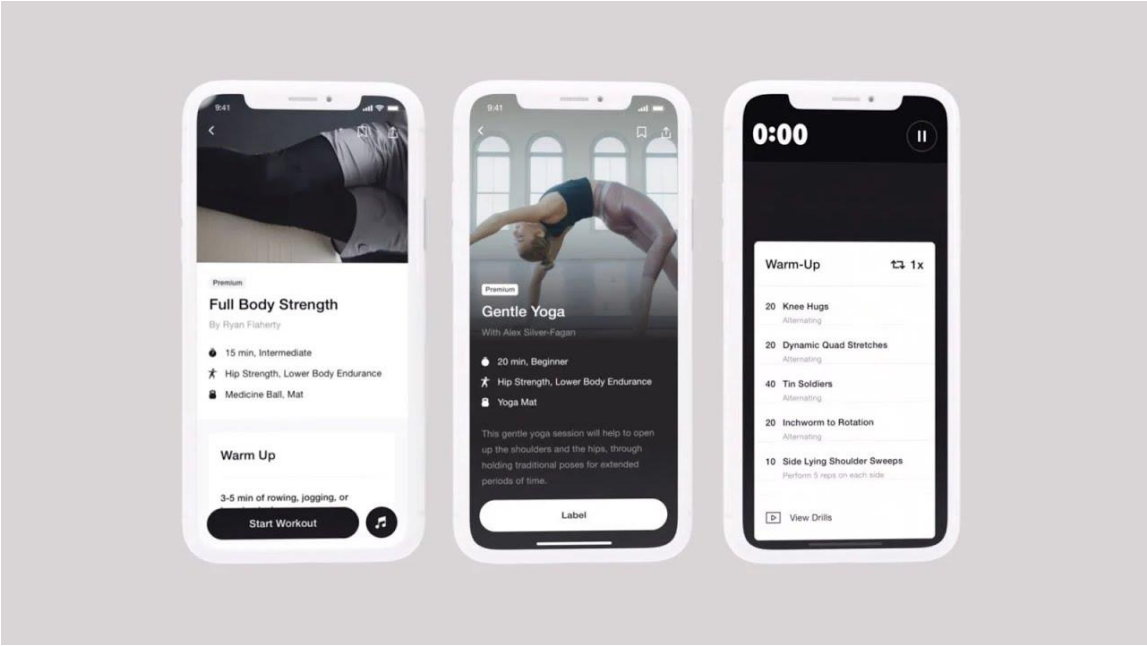


Рисунок 1 - Пример интерфейса NTC

Преимущества Nike Training Club:

— предлагает разнообразные тренировки: силовые, растяжка, кардио, HIIT и йога. Весь контент доступен бесплатно, включая полноформатные программы и советы от профессиональных тренеров. Приложение имеет современный дизайн и хорошо подходит как для начинающих, так и для продвинутых пользователей.

Недостатки Nike Training Club:

— отсутствует возможность покупки курсов, созданных профессиональными тренерами. Невозможно загружать свои тренировочные программы, из-за чего тренеры не могут использовать платформу для заработка или продвижения.

4.2 Adidas Training

Adidas Training – приложение для тренировок в домашних условиях от немецкой компании Adidas.

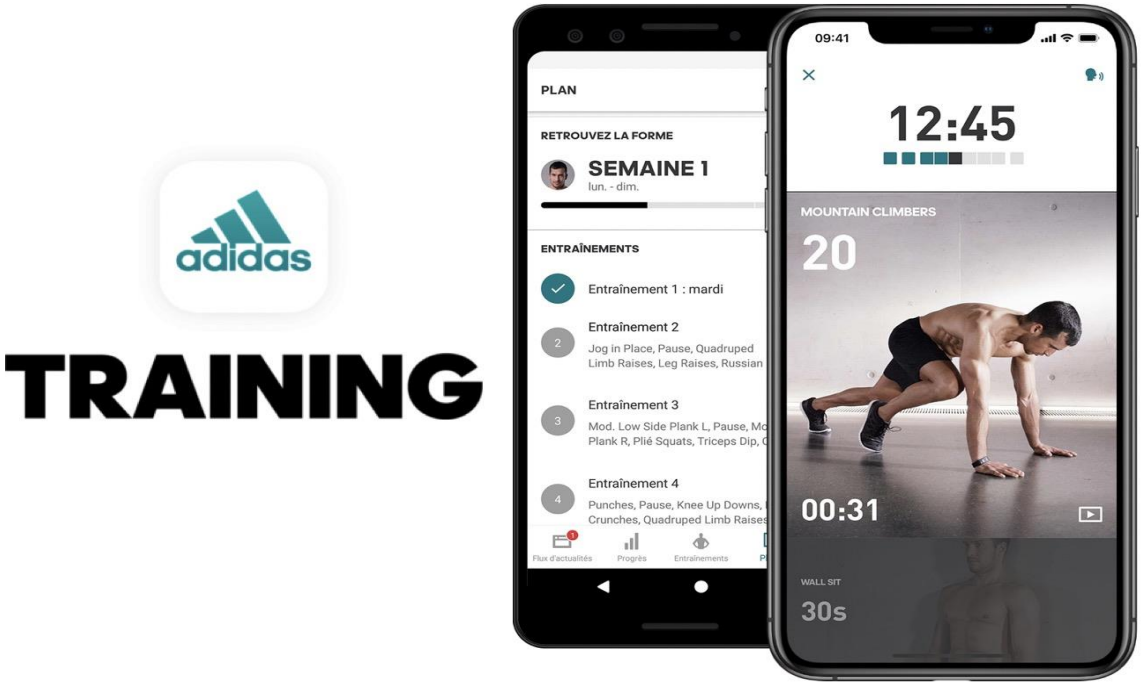


Рисунок 2 - пример интерфейса AdTr

Преимущества Adidas Training:

— приложение предлагает персонализированные планы, основанные на целях пользователя и его уровне подготовки. Хорошо интегрируется с другими сервисами (например, с приложениями по отслеживанию питания и сна). Есть мотивационные челленджи и сообщество единомышленников.

Недостатки Adidas Training:

— несмотря на персонализацию, тренировки, по сути, универсальные, без участия конкретных тренеров. Отсутствует возможность для специалистов размещать свои видеокурсы или строить клиентскую базу внутри приложения.

4.3 Fitify

Fitify – это приложение для тренировок, которое создаёт индивидуальные программы на основе личных целей и опыта в фитнесе.

Рисунок 3 - пример интерфейса Fitify

Преимущества Fitify:

— подходит для тренировок как с оборудованием, так и без него. Работает даже без подключения к интернету. Содержит разнообразные комплексы на все группы мышц.

Недостатки Fitify:

— не предполагает наличие тренеров, отсутствует раздел с видеокурсами и структура обучения. Нет элемента общения или возможности делиться собственными программами.

4.4 Freeletics

Freeletics – это фитнес-приложение, выпущенное Freeletics GmbH в Мюнхене.

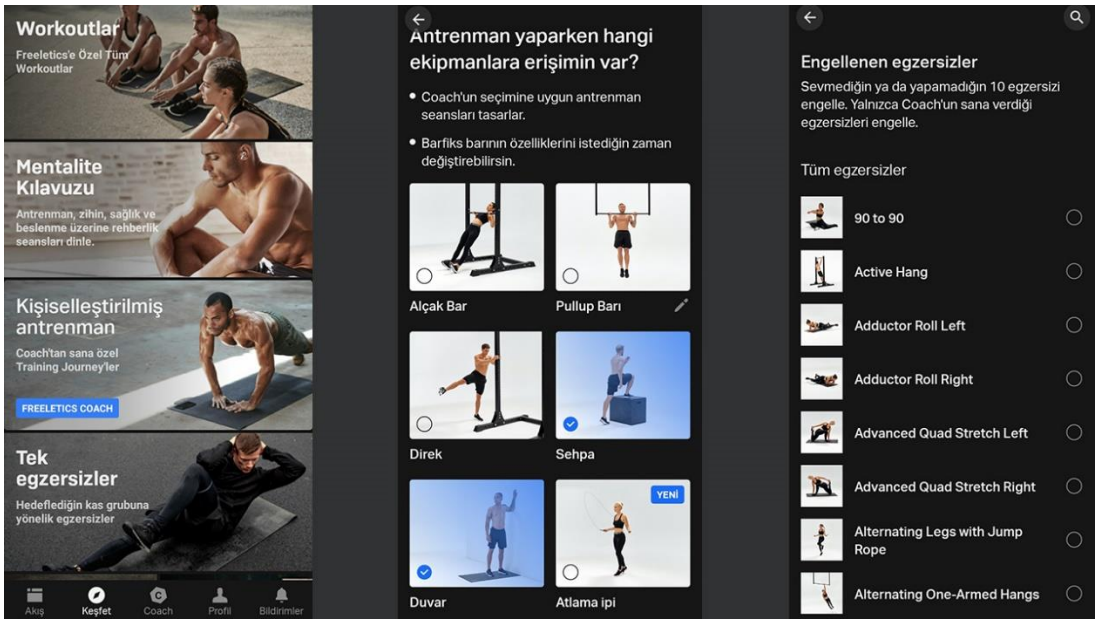


Рисунок 4 - пример интерфейса freeletics

Преимущества Freeletics:

— уникальность в использовании искусственного интеллекта для подбора тренировок. Алгоритм адаптирует программу в зависимости от прогресса пользователя. Подходит для самостоятельных занятий дома и в зале. Присутствует элемент геймификации.

Недостатки Freeletics:

— большая часть функционала доступна только по подписке. Нет возможности выбора курсов от конкретных специалистов. Приложение не предоставляет платформу для загрузки авторских тренировок.

4.5 FitOn

FitOn – цифровая фитнес-платформа, которая предлагает бесплатные занятия, охватывающих разные виды тренировок: кардио, силовые, йогу, пилатес и другие.

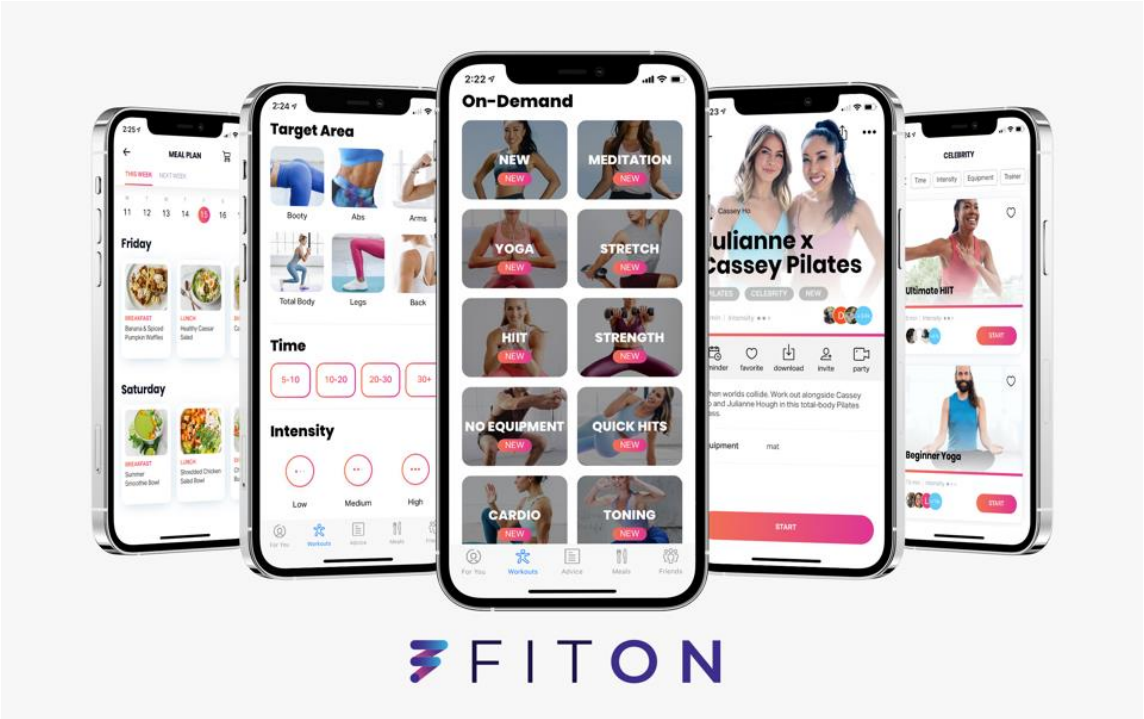


Рисунок 5 - пример интерфейса FitOn

Преимущества FitOn:

— предлагает тренировки с популярными тренерами и знаменитостями. Бесплатный доступ к большому количеству видеотренировок, включая кардио, силовые, йогу, медитации. Подходит для занятий дома без оборудования.

Недостатки FitOn:

— контент не структурирован в полноценные курсы. Пользователь не может выбрать конкретного тренера для прохождения полной программы. Также отсутствует возможность для специалистов выкладывать собственные курсы.

Эти сервисы ориентированы на широкий круг пользователей, но они не предполагают участия конкретных тренеров, в отличии от «Trainova».

Согласно анализу, обладают следующими преимуществами:

— большой выбор тренировок различного уровня сложности;

— высокое качество визуального контента и видеоинструкций;

Тем не менее, «Trainova» включает в себя функциональность, которую конкуренты реализуют частично или не реализуют вовсе:

— возможность покупки видеокурсов у конкретного тренера;

— платформа для тренеров, где они могут размещать и продавать свои программы;

— двойная модель использования: как универсальные тренировки (Блок А), так и персонализированные курсы (Блок Б);

— отсутствие навязанных подписок — пользователь сам решает, чьи курсы приобретать.

Таким образом, приложение «Trainova» интегрирует в себя лучшие черты существующих решений, дополняя платформу уникальной системой продажи тренировочных программ, где тренеры публикуют курсы, а пользователи приобретают.

5 Функциональные требования

Для создания тренировок, которые приложение будет предоставлять пользователю бесплатно, необходим как минимум 1 пользователь от заказчика, который будет выполнять роль администратора.

Для взаимодействия с данной системой выделяют следующие виды пользователей:

— незарегистрированный пользователь;

— зарегистрированный пользователь;

— пользователь с подпиской;

— администратор.

5.1 Незарегистрированный пользователь

Неавторизованный пользователь имеет возможность в процессе взаимодействия с данной системой выполнять следующие действия:

— просматривать тренировки, но без возможности начать заниматься;

— зарегистрироваться.

5.2 Зарегистрированный пользователь

Зарегистрированный пользователь имеет возможность в процессе взаимодействия с данной системой выполнять следующие действия:

— возможности неавторизованного пользователя (кроме регистрации);

— заниматься по бесплатным тренировкам, предоставленными платформой;

— фильтровать тренировки согласно своим потребностям;

— смотреть бесплатные тренировки, предоставленные тренерами;

— отслеживать частоту тренировок и свой вес на графике;

— купить подписку на курс тренировок понравившегося тренера;

— комментировать и ставить оценку тренировкам от тренеров;

— возможность создавать через конструктор свои курсы тренировок;

— изменение личных данных через профиль.

5.3 Пользователь с подпиской

Пользователь с подпиской имеет возможность в процессе взаимодействия с данной системой выполнять следующие действия:

— всё тоже самое что и пользователь без подписки.

— полный доступ к курсу тренировок, на который оформлена подписка.

5.4 Администратор

Администратор имеет возможность в процессе взаимодействия с данной системой выполнять следующие действия:

— создавать тренировки в специальном конструкторе, которые после будут видны пользователям, как бесплатные тренировки от приложения;

— весь функционал, что и у зарегистрированного пользователя;

— доступ к просмотру всех платных курсов тренировок от тренеров.

6 Нефункциональные требования

6.1 Требования к системе в целом

6.1.1 Требования к структуре

Пользователь взаимодействует с серверной частью PWA-приложения через клиентское приложение (PWA), а сервер, используя REST API, возвращает пользователю необходимые данные. Администратор взаимодействует с серверной частью PWA-приложения через клиентское приложение.

Серверная часть приложения включает в себя следующие микросервисы:

— сервис авторизации и регистрации;

— сервис курсов от тренеров (позволяет получать, создавать, редактировать тренировки и курсы);

— сервис тренировок от приложения (позволяет получать, создавать, редактировать тренировки);

— сервис управления профилем (имя, фамилия, пароли, способы оплаты, восстановление паролей, подписки, аватарки);

— сервис генерации мотивационной речи на основе LLM DeepSeek R1-7B и Ollama;

— сервис комментариев для уроков.

6.1.2 Требования к защите информации от несанкционированного доступа

Система должна обеспечивать безопасность данных пользователей и включать в себя следующие механизмы защиты:

— Bearer Authentication с JWT (JSON Web Tokens): Использование стандартного механизма аутентификации Bearer с JWT токенами для безопасной аутентификации и авторизации пользователей;

— безопасное хранение и передачу учетных данных;

— поддержку access и refresh токенов;

— валидацию JWT с проверкой срока действия;

— возможность отзыва токенов через blacklist;

— версионирование токенов при смене пароля;

— хеширование паролей: хранение паролей пользователей в базе данных в хешированном виде с использованием алгоритма bcrypt;

— HTTPS: Обязательное использование защищенного протокола HTTPS для всех взаимодействий с API;

— CORS (Cross Origin Resource Sharing): Настройка политики CORS для предотвращения несанкционированных межсайтовых запросов;

— валидация данных: Строгая валидация всех входящих данных на стороне сервера с использованием Pydantic моделей, что является встроенной функцией FastAPI.

6.1.3 Требования к аутентификации

— в системе должна быть реализована идентификация и проверка доступа при входе в систему по логину/почте и паролю длиной не менее 8 символов;

— система защиты должна подвергать проверке подлинность идентификации – осуществлять аутентификацию;

— система не должна предоставлять доступ к защищённым данным неавторизованным пользователям;

— система защиты должна обладать способностью надёжно связывать полученную идентификацию со всеми действиями данного пользователя;

6.1.4 Требование к патентной чистоте

Проект не должен нарушать никаких патентных прав и лицензий. Все используемые технологии и библиотеки должны иметь соответствующие лицензии, разрешающие их использование в данном проекте.

6.1.5 Требования к масштабируемости и открытости

Проект должен предоставлять возможность добавлять новую функциональность с минимальным изменением существующего кода. Микросервисная архитектура должна обеспечивать независимое масштабирование отдельных компонентов системы в зависимости от нагрузки.

6.1.6 Обработка ошибок

В случае возникновения ошибок пользователь должен получать соответствующие сообщение об ошибке. Приложение должно поддерживать обработку следующих основных ошибок:

— некорректный ввод данных;

— системный сбой;

— ошибки авторизации и аутентификации;

— ошибки при загрузке контента;

— ошибки при работе с видео материалами.

— локального запуска крупных языковых моделей (LLM).

7 Средства реализации

Для реализации приложения использовались следующие средства:

— Python;

— FastAPI;

— СУБД PostgreSQL;

— Docker;

— Nginx;

— Ollama;

— Swagger;

— React;

— Next.js;

— TypeScript;

Выбор указанных технологий продиктован следующими преимуществами:

Для FastAPI:

— высокая производительность и скорость отклика;

— асинхронность и лёгкость масштабирования;

— простота интеграции с Swagger/OpenAPI для документирования.[2]

Для PostgreSQL:

— надёжность и поддержка транзакций;

— масштабируемость;

— гибкая работа с JSON, массивами и пользовательскими типами.[3]

Для Docker:

— удобная упаковка микросервисов и развёртывание в любой среде;

— лёгкое масштабирование и поддержка CI/CD.[4]

8 Реализация серверной части приложения

Для реализации серверной части «Trainova» использован фреймворк FastAPI, работающий поверх асинхронного стека Python 3.11. В качестве архитектурного шаблона выбрана Onion (слоистая) архитектура – разновидность Clean Architecture, целиком отделяющая бизнес-логику от технических деталей.

Onion-подход делит код на четыре концентрических слоя.

1. Domain (ядро) – неизменяемые бизнес-модели и инварианты.
2. Application – use-case-классы (сервисы), описывающие сценарии работы системы.
3. Infrastructure – адаптеры (репозитории asyncpg, почтовый клиент, LLM-клиент).
4. API – роутеры FastAPI, преобразующие HTTP запросы в вызовы Application-слоя.

Основные преимущества Onion архитектуры:

— чёткое разделение ответственности между бизнес-логикой и техническими деталями;

— читаемость и поддержка кода;

— возможность лёгкой замены инфраструктурных компонентов без изменения ядра;

— глубокая тестируемость – Domain и Application проверяются юнит-тестами без запуска сервера;

— масштабируемость: каждый слой можно развивать независимо, а микросервисы развёртываются автономно.

Таким образом, использование FastAPI вместе с Onion-архитектурой обеспечивает высокую скорость разработки, строгую изоляцию бизнес-правил, простое масштабирование микросервисов и удобную поддержку кода.

8.1 Слой доступа к данным

Используется библиотека asyncpg, которая обеспечивает прямое взаимодействие с PostgreSQL. Для каждой сущности определён собственный репозиторий, реализующий контракт интерфейса из слоя Domain.

Репозитории предоставляют методы для основных операций доступа к данным, таких как создание, чтение, обновление и удаление (CRUD), а также специализированные запросы: поиск по идентификатору, фильтрация по различным критериям.

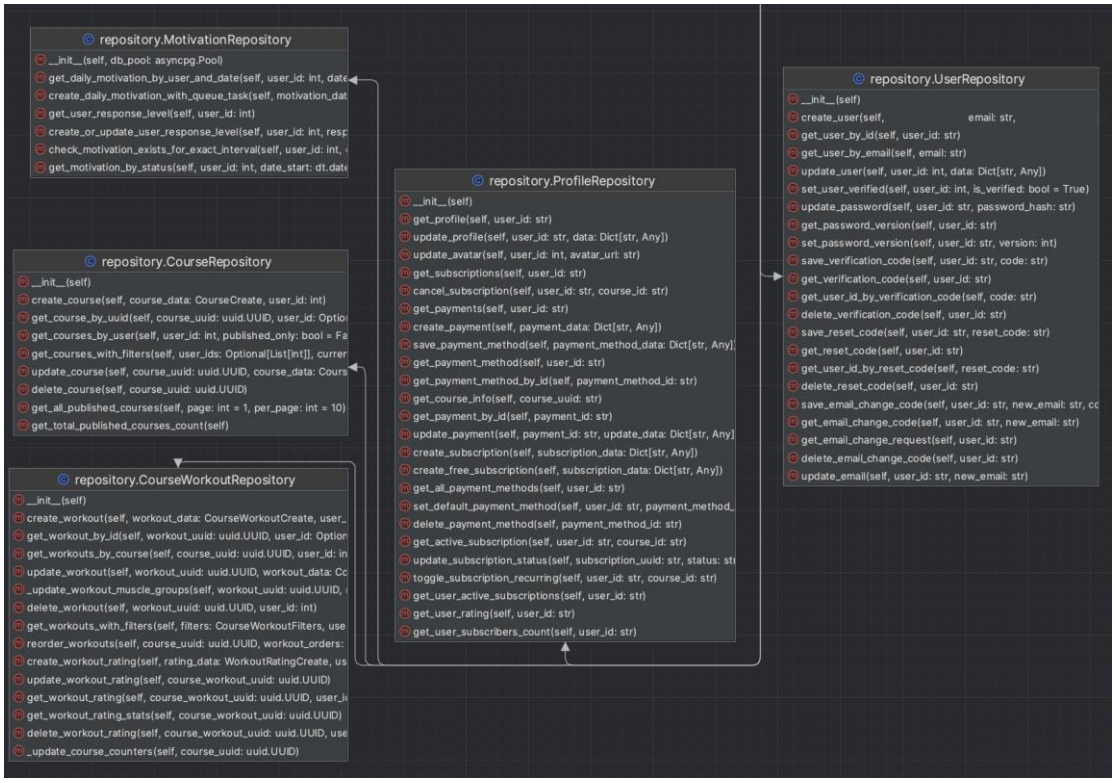


Рисунок 6 - Схема репозиториев

8.2 Слой контроллеров (роутеров)

Компонент API представлен набором роутеров FastAPI. Каждый микросервис публикует Swagger-документацию, формируя REST интерфейс для front-end-PWA.

Для ключевых сущностей реализованы следующие роутеры:

— api/auth/router.py;

— api/workout/router.py;

— api/course/router.py;

— api/course/course_workout_routes.py;

— api/comment/router.py;

— api/profile/router.py.

— api/motivation/router.py.

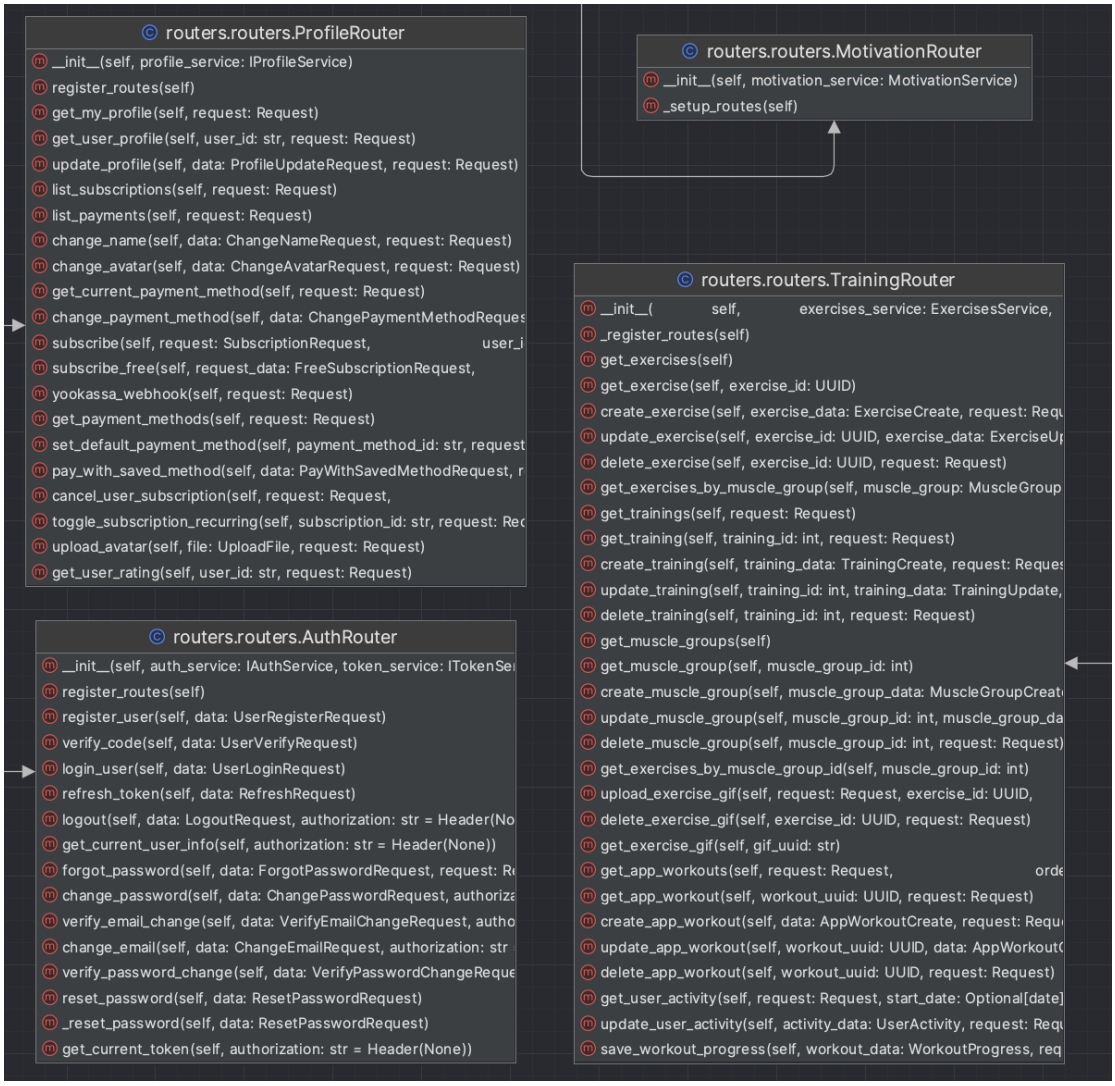


Рисунок 7 - Схема роутеров.

8.3 Слой моделей

Доменные сущности описаны неизменяемыми Pydantic-классами. Основной набор:

— User;

— Profile;

— Course;

— CourseWorkout;

— WorkoutRating;

— DailyMotivation;

— NeuroGenerationQueue;

— Subscription, Payment.

Модели содержат только бизнес-атрибуты и минимум валидации, что обеспечивает тестирование без подключения к Infrastructure-слою.

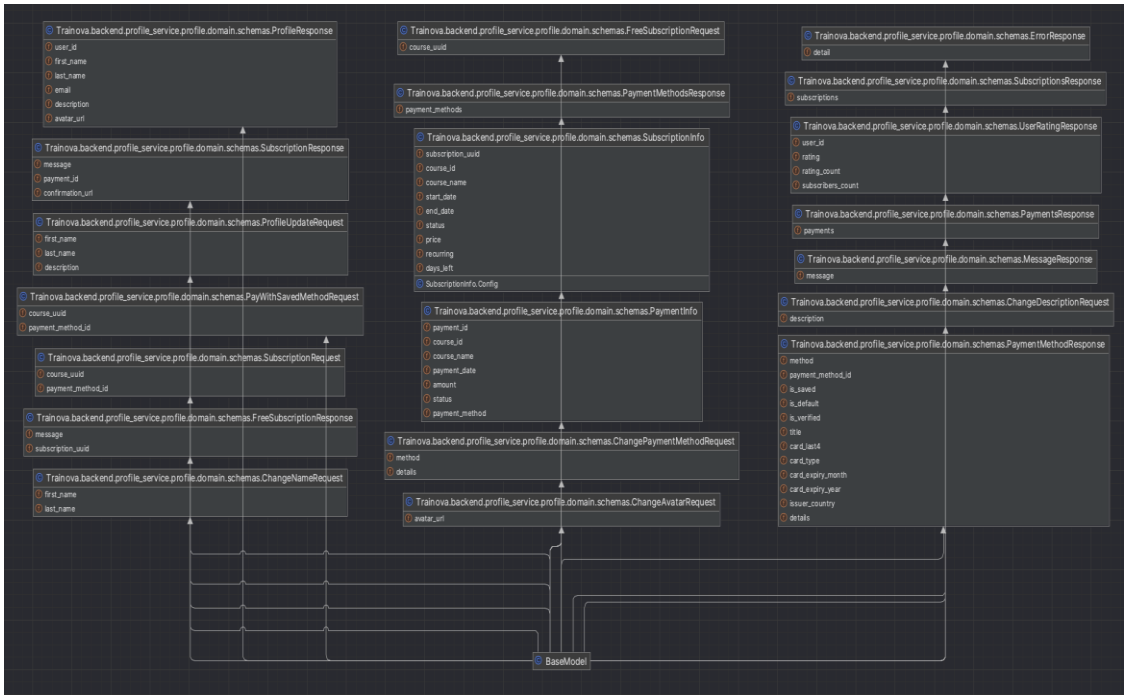


Рисунок 8 - Реализация моделей

8.4 Сервис-слой (Application)

Сервис слой служит посредником между слоем контроллеров и слоем доступа к данным, обеспечивая абстракцию бизнес-логики. В большинстве случаев методы этого слоя просто передают аргументы, полученные от контроллера, в соответствующие методы репозитория и возвращают ответ обратно в контроллер.

Кроме того, сервисы могут формировать объекты, на основе полученных аргументов, передав их в репозиторий для выполнения операций с данными.

Это позволяет изолировать бизнес-логику от деталей реализации доступа к данным и упрощает тестирование. Ниже представлена схема сервисов.

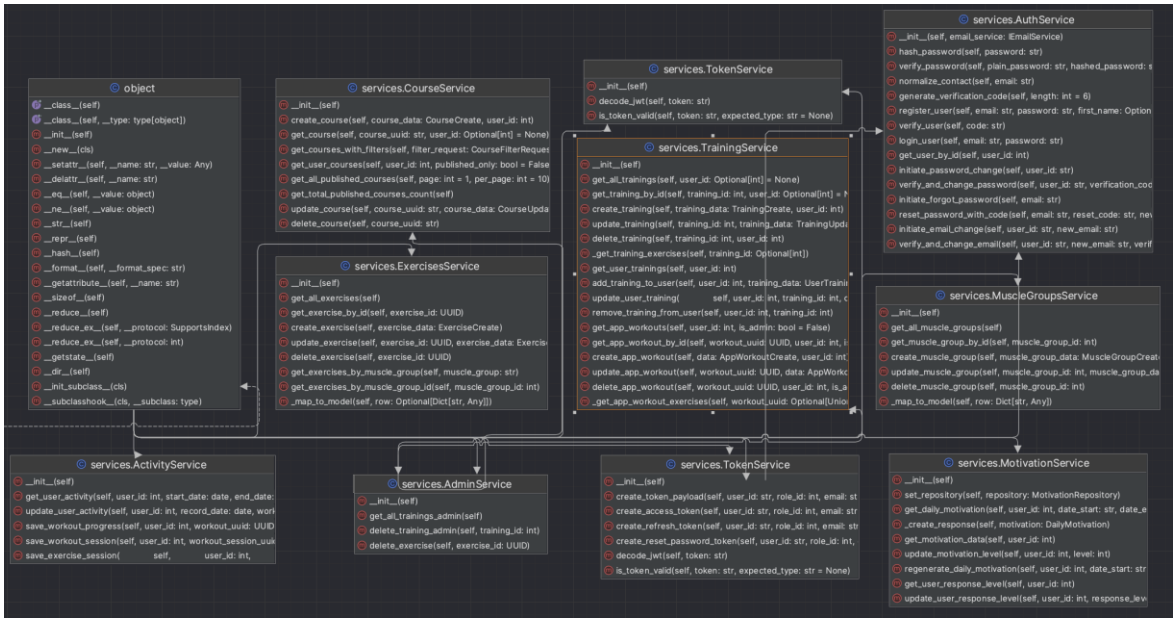


Рисунок 9 - Схема сервисов

8.5 Механика работы приложения

Для полноценного функционирования задействованы сервисы:

— AuthService – регистрация, вход, refresh-/access-JWT, отзыв токенов;

— ProfileService – содержит методы для управления (CRUD) профилями, загрузкой аватара, подписками, данными платежных карт и личной информацией.

— WorkoutService– содержит методы управления (CRUD) упражнениями, группами мышц, тренировками, активностью пользователей.

— CourseService – содержит методы управления (CRUD) тренировками, курсами, рейтингом;

— CommentService– CRUD комментариев;

— MotivationService – генерация мотивационной речи на основе анализа данных активности пользователя (LLM Olama + DeepSeek R1);

Методы Application-сервиса выполняют следующие шаги:

— получение DTO из роутера.

— валидация входных данных Pydantic.

— вызов репозитория (или нескольких) для чтения/записи.

— формирование ответа и возврат в роутер.

9 Реализация клиентской части приложение

Для реализации клиентской части приложения используется фреймворк Next.js с App Router в сочетании с React. Приложение основано на Feature-based архитектуре, которая разделяет приложение на функциональные модули и обеспечивает четкое разделение ответственности между слоями.

Данная архитектура включает в себя четыре основных слоя: Представление (UI), Бизнес-логика, Доступ к данным и Маршрутизация.

Представление отвечает за отображение интерфейса пользователя и реализовано с помощью компонентов React/Next.js, расположенных в директориях components.

Бизнес-логика содержится в хуках (hooks) и контекстах (contexts), что обеспечивает переиспользуемость кода и отделение логики от представления.

Доступ к данным осуществляется через сервисный слой (services/api.ts), который предоставляет типизированные интерфейсы для взаимодействия с API.

Маршрутизация реализована с помощью App Router Next.js, что позволяет организовать структуру приложения на основе файловой системы.

Для управления состоянием приложения используется React Context API, в частности AuthContext для глобального состояния аутентификации, а локальное состояние управляется через React hooks (useState, useEffect).

Взаимодействие с API реализовано через централизованный сервисный слой с модульным подходом и разделением по доменам (auth, profile, workout, comment, motivation, course.)

Стилизация и UI компоненты реализованы с использованием Material UI (MUI) как основной библиотеки компонентов, а также Tailwind CSS.

Кастомная тема (theme.ts) обеспечивает управление стилями и поддерживает единообразие интерфейса.

Навигация в приложении осуществляется с помощью компонентов маршрутизации Next.js и навигационной панели, общим для всех страниц является макет (layout). Пользователь может переходить между различными разделами приложения, такими как главный экран, тренировки, курсы и профиль.

9.1 Загрузочный экран

Загрузочный экран содержит логотип «Trainova», надпись: «Добро пожаловать в Trainova!» и кнопку для перехода на главную страницу

Рисунок 10 - Загрузочный экран

9.2 Раздел «Неавторизованный пользователь»

9.2.1 Страница тренировок для неавторизованного пользователя

После нажатия на кнопку начать, неавторизованный пользователь попадает на раздел «Блок тренировок», а также посмотреть список курсов от тренеров в разделе «Блок курсов». При нажатии на тренировку, раздел «Главная» или «Личный кабинет» неавторизованный пользователь попадает на страницу входа.

9.3 Раздел «Авторизации»

9.3.1 Страница входа

Данное окно создано для регистрации пользователя. Оно содержит поле ввода логина и пароля пользователя.

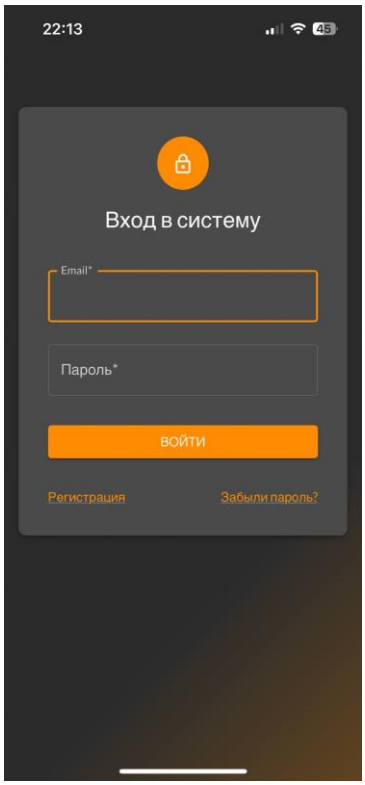


Рисунок 11 - Вход в систему

9.3.2 Страница регистрации

Страница регистрации содержит поле ввода почты, имени, фамилии пароля, подтверждения пароля, а также кнопку зарегистрироваться с подтверждением почты.

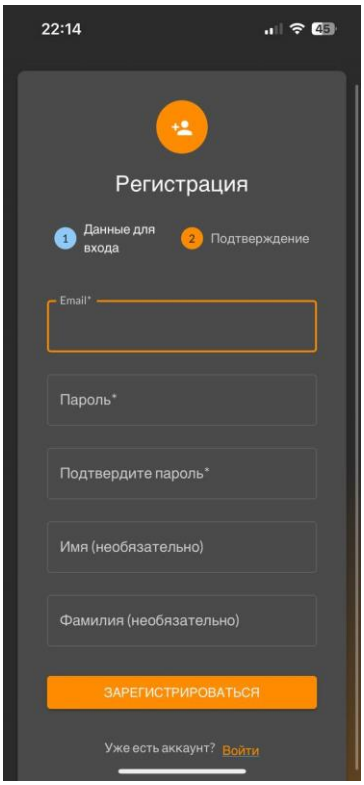


Рисунок 12 - Регистрация пользователя

9.3.3 Страница восстановления пароля

Страница восстановления пароля содержит поле ввода почты, на которую приходит ссылка для восстановления пароля.

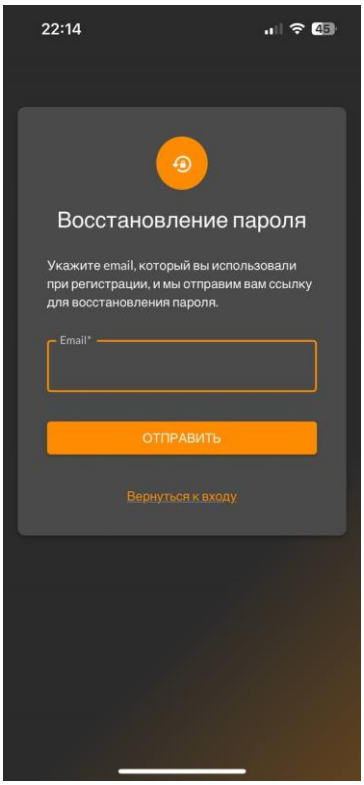


Рисунок 13 - Восстановление пароля

9.4 Раздел навигационной панели «Главная»

9.4.1 Страница «Главная»

Первая страница, куда попадает пользователь после загрузочного экрана – «Главная».

На данной странице представлены следующие компоненты:

— мотивирующие сообщение и советы от нейросети;

— график с изменениями веса и активностью пользователя;

— все купленные тренировки и последние пройденные от приложения.

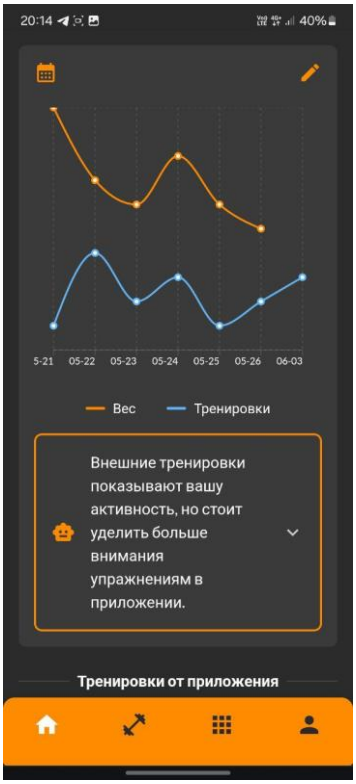


Рисунок 14 - Главный экран

9.4.2 Страница редактирования веса

В данном поле можно ввести свой текущий вес, а также изменять его. Помимо этого, пользователь может отмечать свою активность. Все данные отображаются на графике.

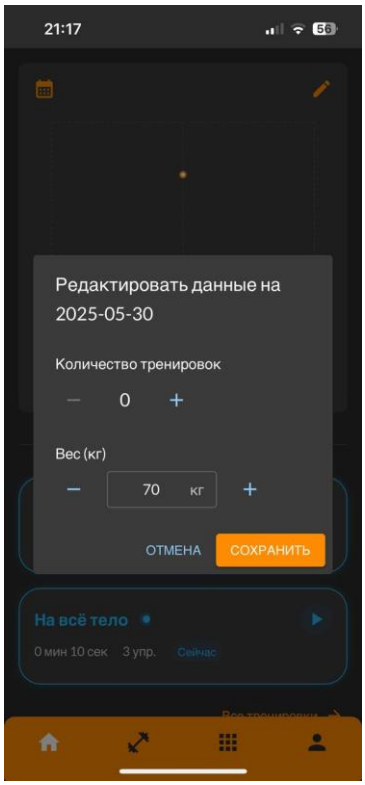


Рисунок 15 - Редактирование веса

9.4.3 Страница редактирования даты

Здесь можно выбрать период, чтобы посмотреть свою активность за определенный период

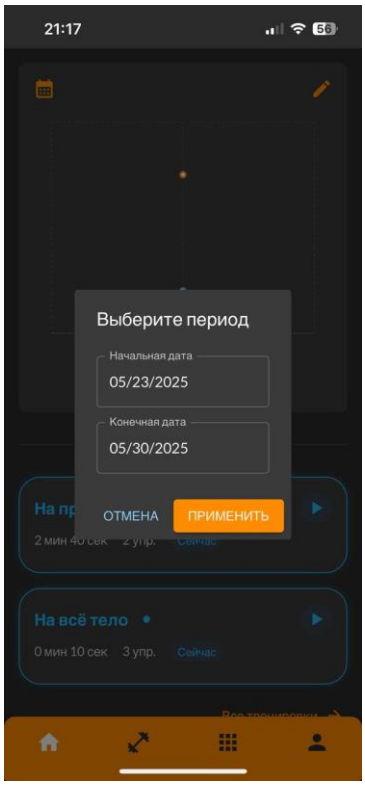


Рисунок 16 - Календарь

9.5 Раздел навигационной панели «Блок тренировки»

9.5.1 Страница настройки тренировки для администратора

На этой странице представлена настройка тренировки. В этом окне администратор может добавлять или удалять упражнения из тренировки и указывать количество повторений время выполнения, а также дать название и, при необходимости, полностью удалить тренировку;

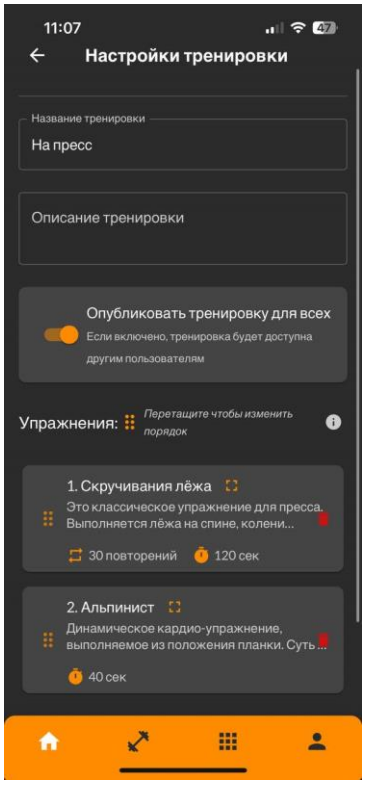


Рисунок 17 - Настройка тренировки

9.5.2 Страница пула упражнений для администратора

Пул упражнений. В данном окне администратор может посмотреть все упражнения на разные группы мышц, а также добавить новое упражнение;

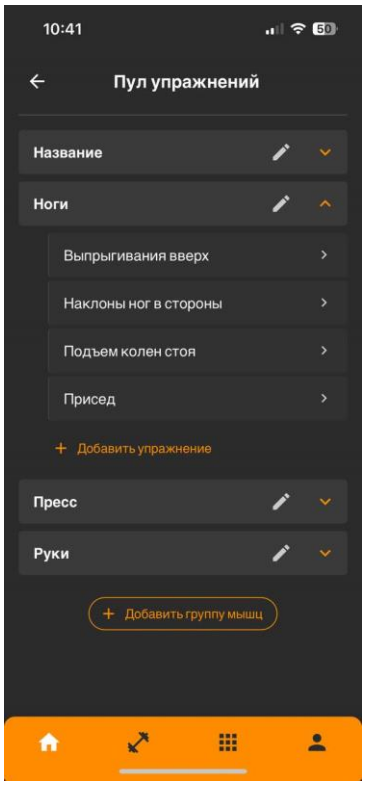


Рисунок 18 - Пул упражнений

9.5.3 Страница добавления упражнения для администратора

Так же администратор может создать тренировку добавить упражнения и указывать количество повторений время выполнения, а также дать название.

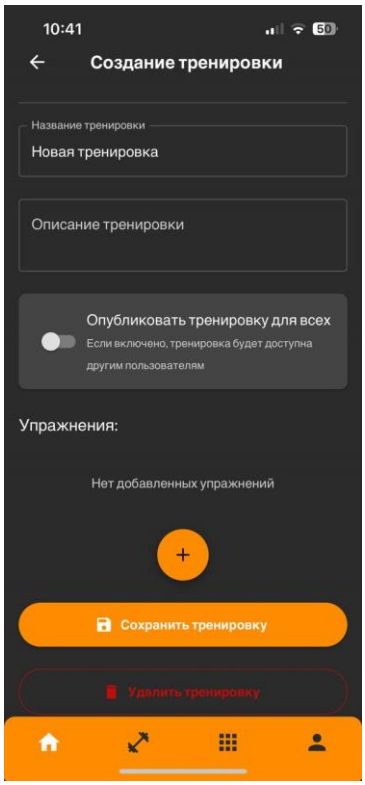


Рисунок 19 - Создание тренировки

9.5.4 Страница блока тренировок, составленных приложением

Тренировки из блока А. Здесь представлены тренировки, составленные приложением.

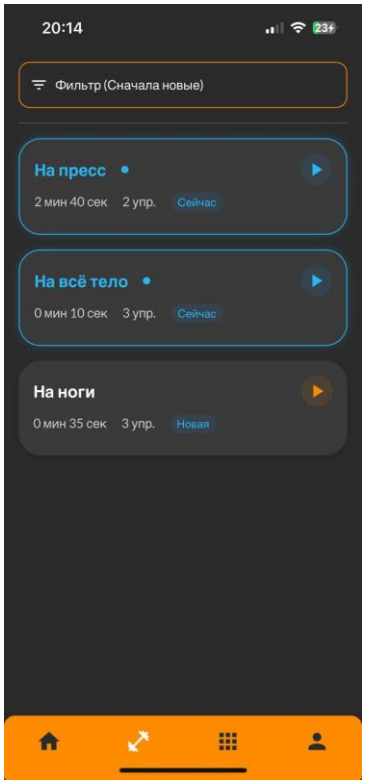


Рисунок 20 - Блок тренировки

9.5.5 Страница тренировки

На страницы тренировки доступно:

— видео упражнения для его визуализации;

— сколько по времени или повторений делать упражнение;

— возможность перехода между упражнениями (Предыдущее/Следующее упражнение);

— оставшееся время тренировки.

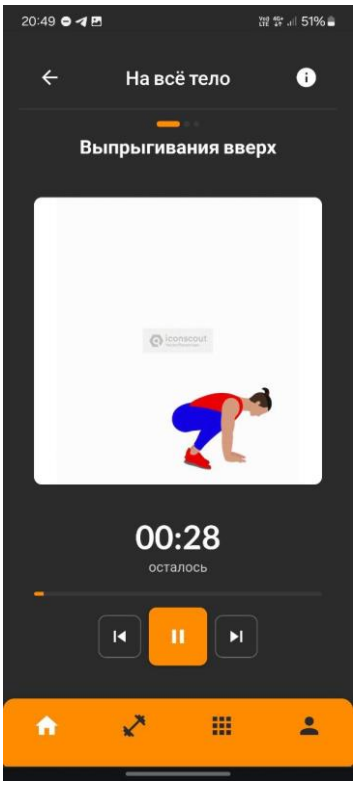


Рисунок 21 - Тренировка

9.5.6 Страница фильтрации тренировок

На данной страницы можно отфильтровать по типам: старые/новые, а также тренировки по нужным группам мышц.

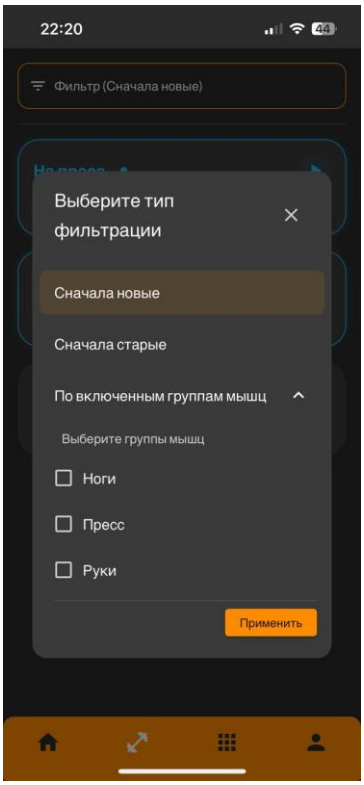


Рисунок 22 - Фильтрация тренировок

9.6 Раздел навигационной панели «Блок курсов»

В данном блоке пользователь может ознакомиться с разными курсами, выложенными тренерами, помимо этого, он может перейти в свой профиль, а также все созданные им курсы.

— курсы от тренеров, где есть название курса, краткое описание, рейтинг, а также включённые группы мышц, помимо этого пользователь может перейти от курса на аккаунт тренера, создавшего этот курс.

— ваши курсы. На данном окне пользователь может просмотреть все созданные им курсы.

9.6.1 Страница с курсами

На этой странице вы можете посмотреть курсы от тренеров, где есть название курса, краткое описание, рейтинг, а также включённые группы мышц, помимо этого пользователь может перейти от курса на аккаунт тренера, создавшего этот курс.

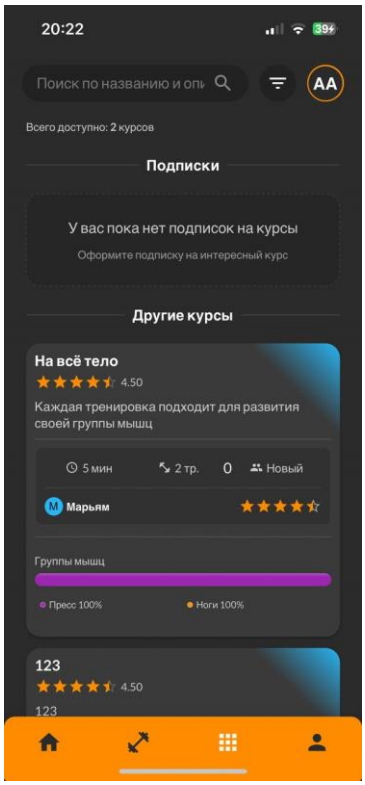


Рисунок 23 - Блок курсов

9.6.2 Страница профиля тренера

На этой странице вы можете посмотреть профиль тренера, увидеть рейтинг, а также посмотреть все курсы, представленные тренером.

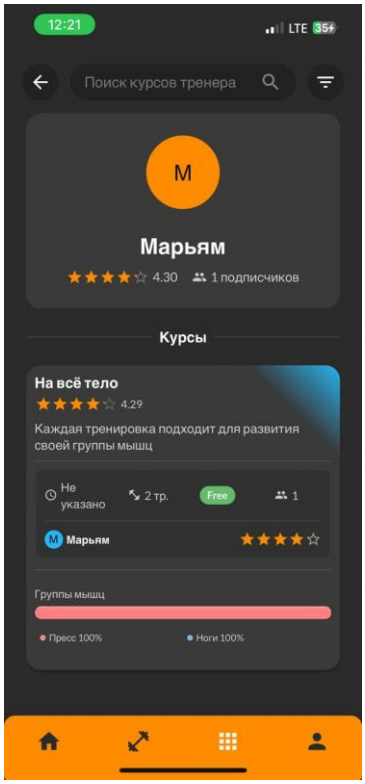


Рисунок 24 - Профиль тренера

9.6.3 Страница фильтра курсов

На данной станице можно пользователь может фильтровать курсы или найти тот, который ему нужен.

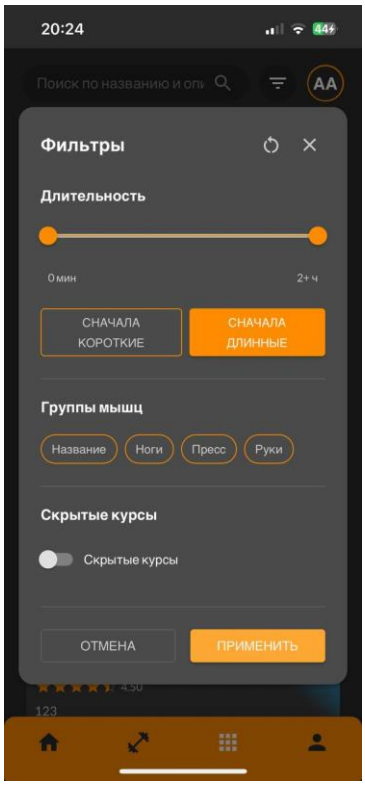


Рисунок 25 - Фильтр курсов

9.6.4 Страница курса

На странице курса есть название курса, краткое описание, рейтинг, а также включённые группы мышц. Перейдя на конкретный курс, созданный ранее, пользователь видит все тренировки, включённые в этот курс, а также у него есть возможность создать новую, для этого он попадает в следующий конструктор, где у него есть возможность указать следующие параметры:

— название тренировки;

— описание урока;

— группы мышц.

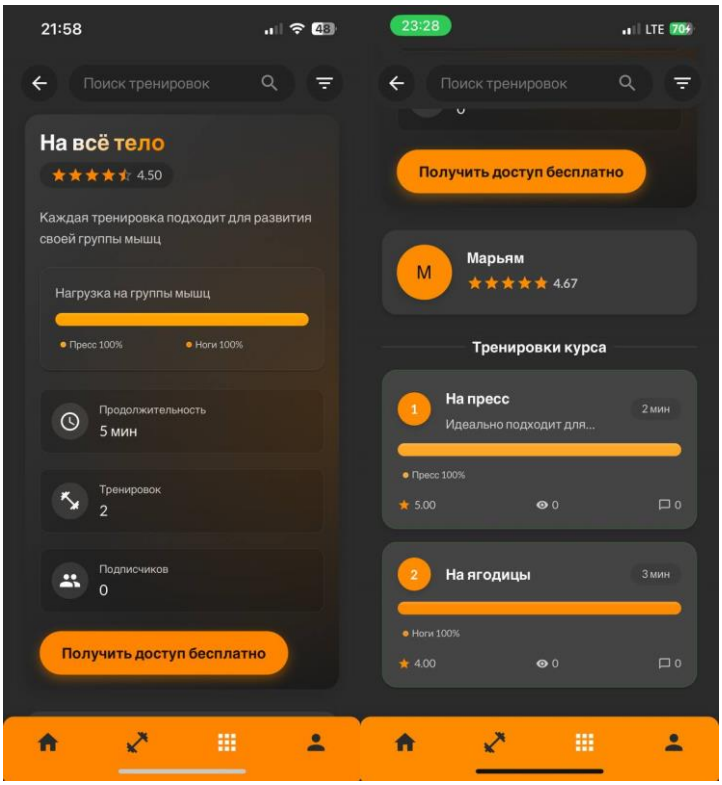


Рисунок 26 - Курс

9.6.5 Страница тренировки из курса

На данной странице доступно видео тренировки, описание, а также возможность читать и оставлять комментарии.

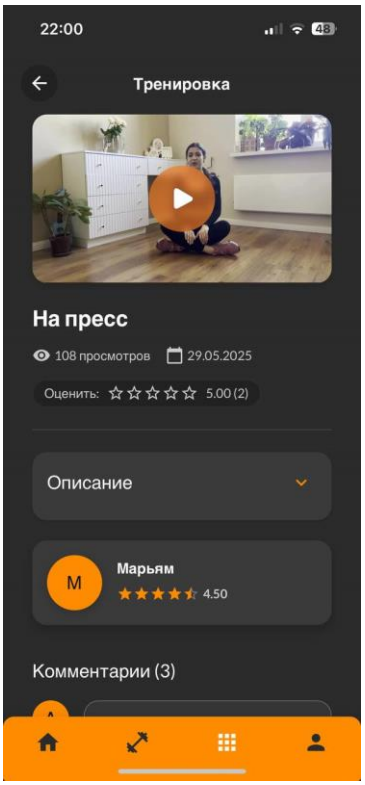


Рисунок 27 - Тренировка из курса

9.6.6 Страница создания курса

На этой странице тренер может создать тренировку. При создании нового курса пользователю необходимо ввести следующие данные:

— название тренировки;

— описание урока;

— настройки группы мышц;

— ссылка на видео;

— опубликовать

— сделать бесплатным/ указать цену;

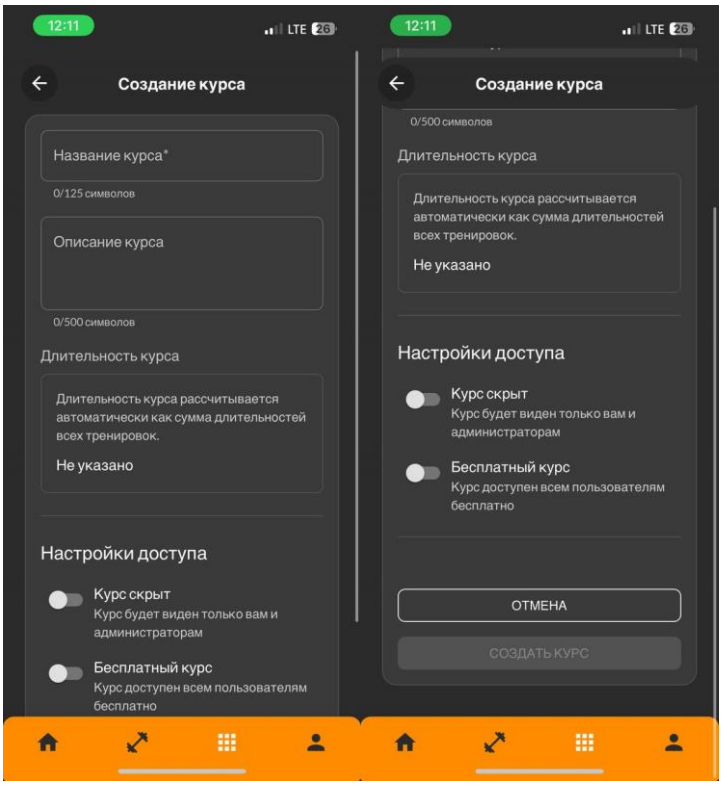


Рисунок 28 - Создание курса

9.6.7 Страница настройки курса

Здесь тренер может отредактировать свой курс, сохранить изменения, а также удалить курс.

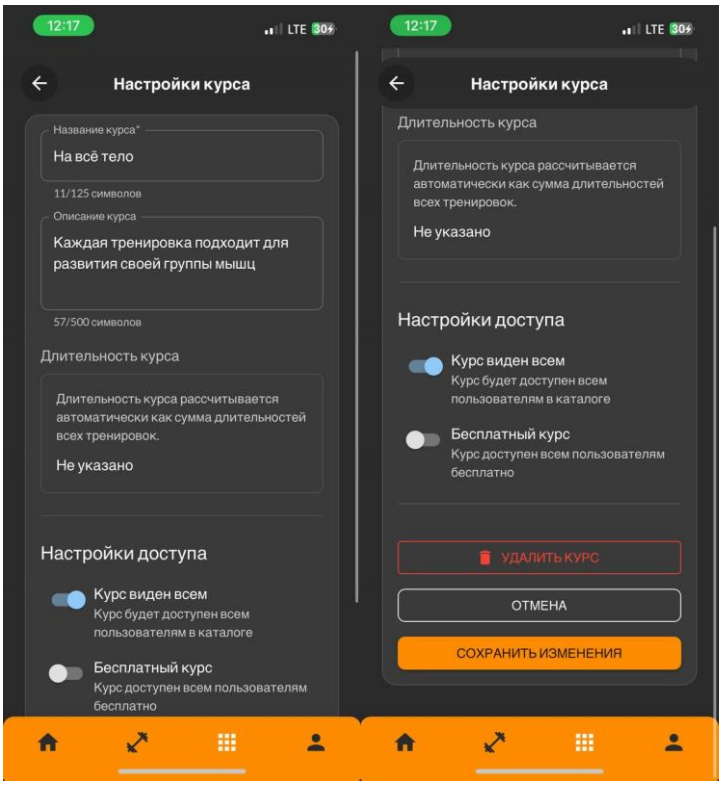


Рисунок 29 - Настройка курса

9.6.8 Страница настройки тренировки из курса

На этой странице тренер может отредактировать свою тренировку.

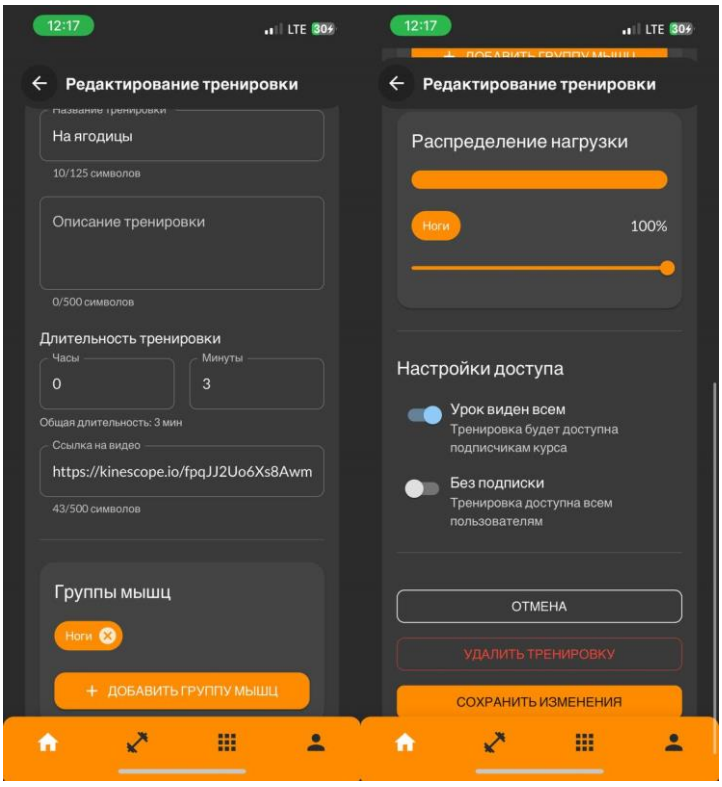


Рисунок 30 - Конструктор курсов

9.7 Раздел навигационной панели «Личный кабинет»

9.7.1 Страница профиля

В данном блоке пользователь может указать/отредактировать свои ФИО, аватар, написать описание профиля, управлять подписками и финансами или же выйти из профиля.

— безопасность. Здесь пользователь может сменить эл.почту , а также поменять пароль;

— управление подписками. Пользователь может отменять подписки на купленные им курсы;

— финансы. Пользователь указывает свои реквизиты для списывания средств. Также здесь реализованы реквизиты для зачисления для пользователей, выложивших хотя бы одну платную тренировку;

— выйти. Выход из аккаунта.

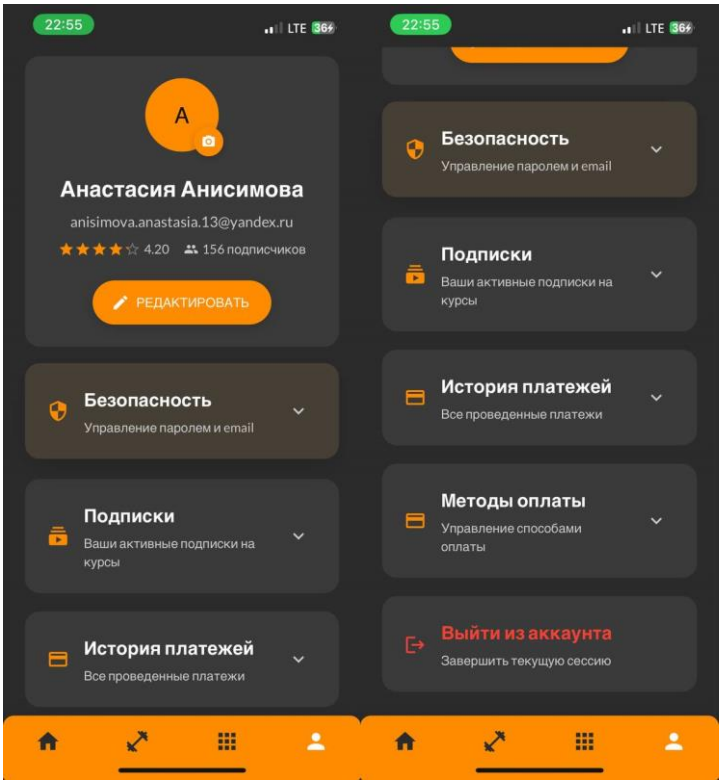


Рисунок 31 - Личный кабинет

9.7.2 Страница редактирования профиля

В данном окне пользователь может изменить имя или фамилию, а также добавить/изменить описание.

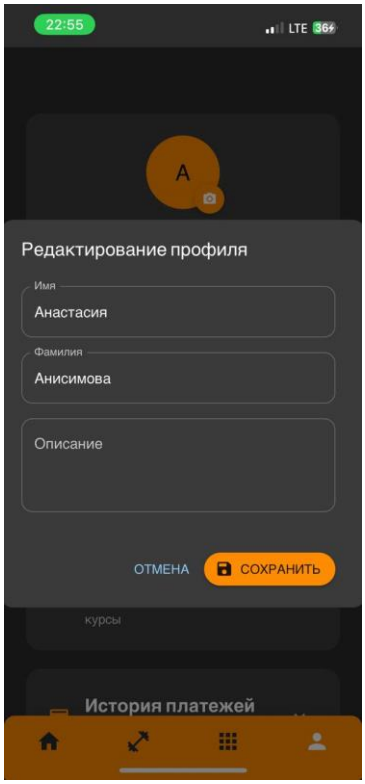


Рисунок 32 - Редактирование профиля

9.8 Описание архитектуры клиентской части

Код разделен на 4 части:

— представление (UI);

— бизнес-логика;

— доступ к данным;

— маршрутизация.

Все слои разделены соответственно в файловой системе:

— для Представления: директория app/components содержит React-компоненты;

— для Бизнес-логики: директории app/hooks и app/contexts содержат логику приложения;

— для Доступа к данным: директория app/services с файлом api.ts для работы с API;

— для Маршрутизации: структура директорий в app и файлы page.tsx.

9.8.1 Слой представления

Отвечает за отображение пользовательского интерфейса и взаимодействие с пользователем. В проекте он реализован с помощью компонентов React, которые используют библиотеку Material UI. Компоненты представления разделены по функциональным областям.

Для управления состоянием компонентов используются React-хуки, такие как useState и useEffect, а для доступа к глобальному состоянию - контексты, созданные с помощью React Context API.

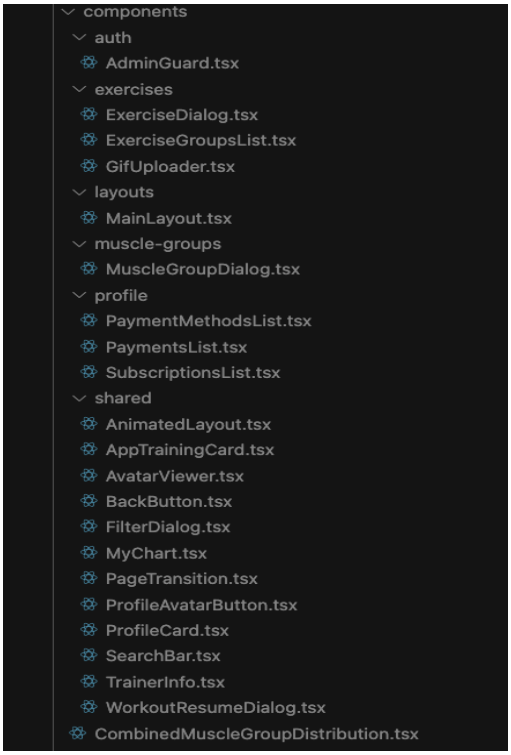


Рисунок 33 - Пример компонента представления

9.8.2 Слой бизнес-логики

Отвечает за обработку данных, управление состоянием и реализацию основных функциональных возможностей. В проекте этот слой реализован с

помощью React-хуков (hooks) и контекстов (contexts), что обеспечивает переиспользуемость кода и отделение логики от представления.

Основные компоненты слоя бизнес-логики включают в себя пользовательские хуки, которые инкапсулируют логику работы с данными, обработку событий и взаимодействие с API.

Хук useAuth содержит всю логику, связанную с аутентификацией пользователя, включая вход, выход, регистрацию и обновление токенов. Контексты React используются для управления глобальным состоянием приложения и обеспечения доступа к данным из различных компонентов.

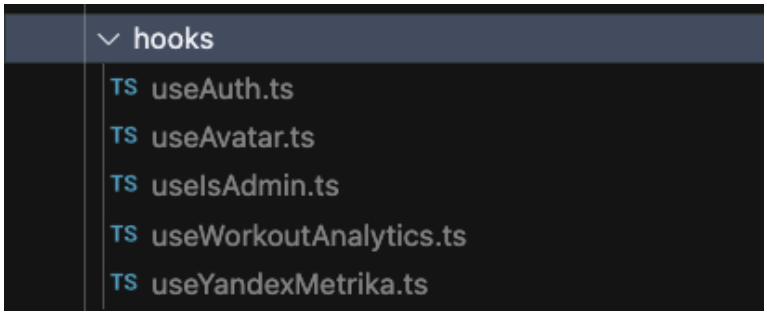


Рисунок 34 - Пример реализации хука бизнес-логики

Бизнес-логика также включает в себя обработку форм, валидацию данных, расчеты (например, распределение нагрузки по группам мышц), форматирование данных для отображения и управление состоянием пользовательского интерфейса. Эта логика изолирована от компонентов представления, что делает код более тестируемым и поддерживаемым.

9.8.3 Слой доступа к данным

Отвечает за взаимодействие с серверным API. В проекте этот слой реализован через сервисный модуль, расположенный в директории app/services.

Основной файл api.ts содержит типизированные интерфейсы для запросов и ответов в формате JSON.

Для обеспечения безопасности и авторизации используется механизм JWT-токенов, которые автоматически добавляются к запросам как Bearer.

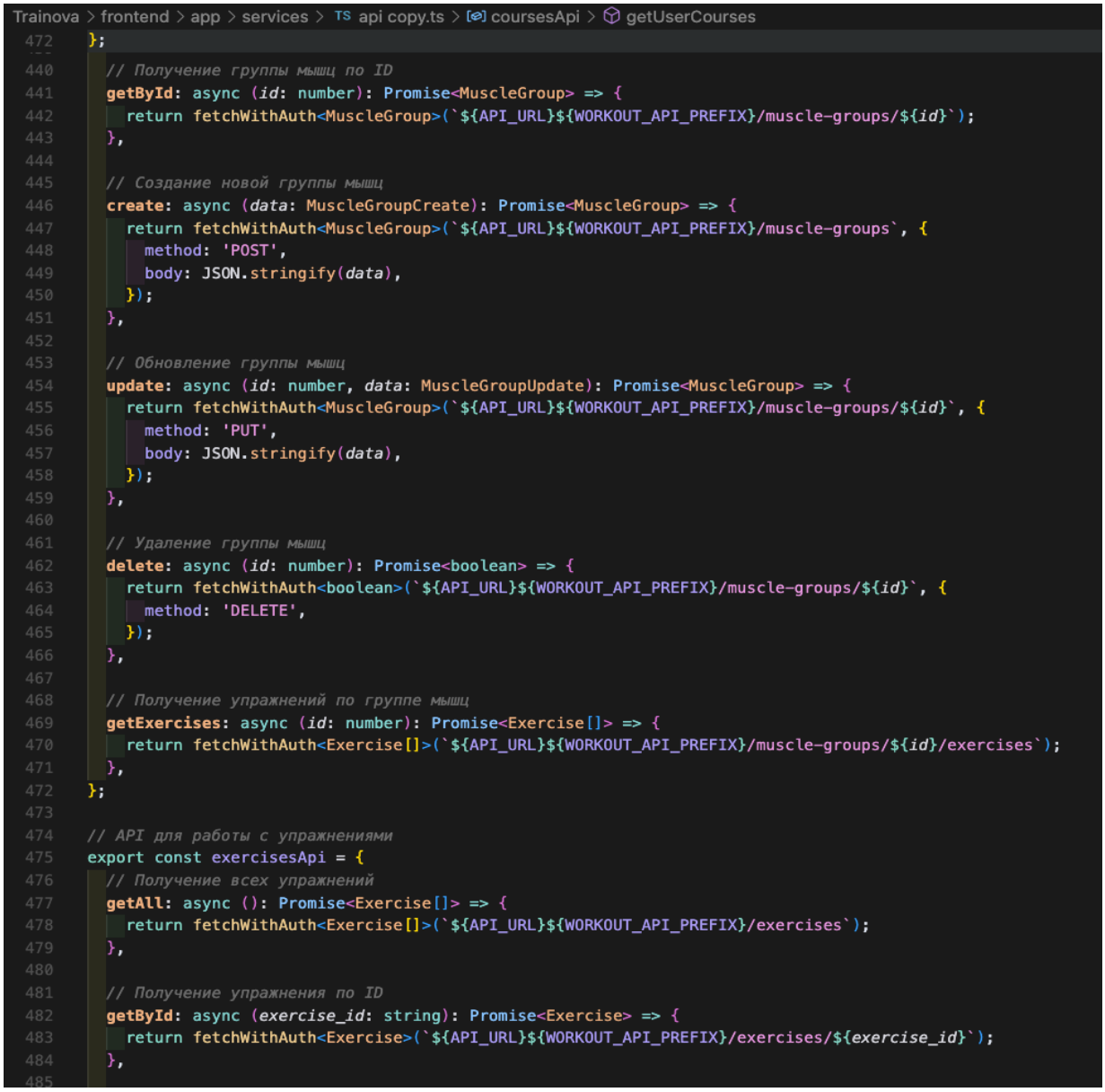


Рисунок 35 - Пример реализации доступа к данным.

Сервисный слой организован по доменам (auth, profile, workout, course, motivation, comment), что обеспечивает модульность и упрощает поддержку кода. Каждый домен содержит набор функций для работы с соответствующими данными.

9.8.4 Слой маршрутизации

Слой маршрутизации отвечает за навигацию между различными страницами приложения и определение структуры URL. В проекте используется система маршрутизации Next.js App Router, которая основана на файловой структуре.

В этом подходе каждая директория внутри папки app соответствует определенному маршруту, а файлы page.tsx определяют содержимое страницы. Такая организация обеспечивает интуитивно понятную структуру проекта, где иерархия файлов напрямую отражает структуру URL.

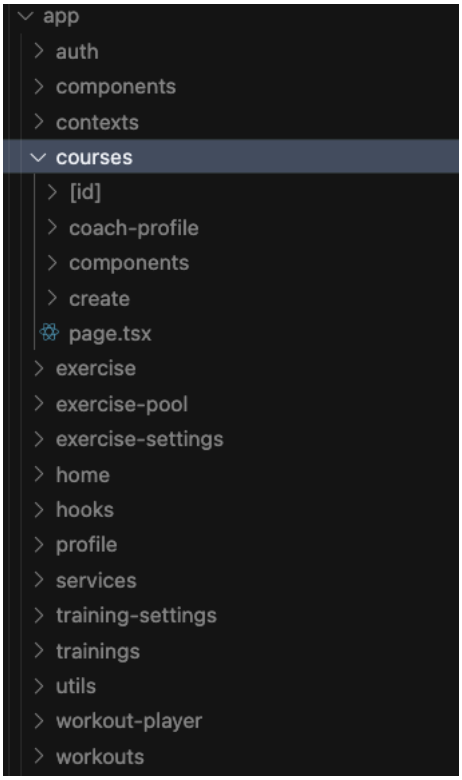


Рисунок 36 - Пример структуры маршрутизации.

Общий макет для всех страниц определен в файле app/layout.tsx, который содержит элементы, присутствующие на каждой странице (навигационное меню, заголовок, подвал). Для динамических маршрутов используются директории с именами в квадратных скобках [id].

10 CI/CD

Для автоматизации процессов сборки, тестирования и развертывания в проекте «Trainova» применяется CI/CD, реализованный через GitHub Actions.

Непрерывная интеграция (CI) запускается при коммитах в ветки main, develop, feature/*, bugfix/*, при этом отслеживаются изменения в соответствующих директориях backend/ и frontend/. В случае изменений в backend/ происходит установка Python-зависимостей, загрузка переменных окружения с удалённого сервера и запуск модульных тестов с использованием pytest для микросервисов. Аналогично, при изменениях в frontend/ происходит установка зависимостей через npm ci, сборка приложения (npm run build) и проверка успешной компиляции.

Фаза непрерывного развертывания (CD) активируется только после успешного CI и при коммитах в ветки main или develop. В зависимости от того, какие файлы были изменены, запускается отдельный сценарий развёртывания. Для фронтенда используется отдельный docker-compose файл, который отвечает исключительно за запуск интерфейсной части, тогда как бэкенд разворачивается через другой docker-compose, включающий в себя все микросервисы. Развёртывание производится через защищённое SSH-соединение, последовательность действий включает остановку текущих контейнеров, пересборку без кэша и запуск в фоновом режиме. Все переменные окружения и ключи доступа хранятся в GitHub Secrets, обеспечивая безопасность конфигурации при развёртывании.

11 Аналитика

К приложению были подключён сервис "Яндекс Метрики", позволяющий собирать аналитику по использованию приложения, собираются следующие важные количественные показатели:

— заходы в приложение;

— начало тренировки от приложения;

— конец тренировки от приложения;

— переключение на следующие\предыдущие упражнение;

— открытие курса тренера;

— подписка на бесплатный курс тренера;

— подписка на платный курс тренера;

— завершение тренировки от тренера;

— открытие страницы с тренировкой от тренера.

Помимо этого, считываются и другие показатели, но были перечисленные основные.

В будущем все собранные данные можно использовать для улучшения качества приложения.

11.1 Воронки

Также для аналитики работы приложения были составлены 4 воронки, а именно:

— прохождение тренировки от приложения;

— прохождение тренировки от тренера;

— оформление подписки на курс;

— оформление бесплатной подписки на курс.

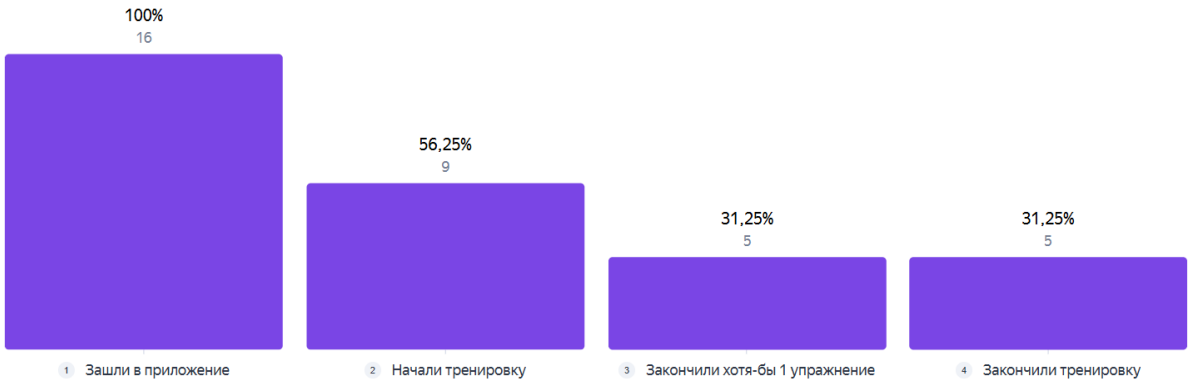


Рисунок 37 - Воронка прохождения тренировки от приложения

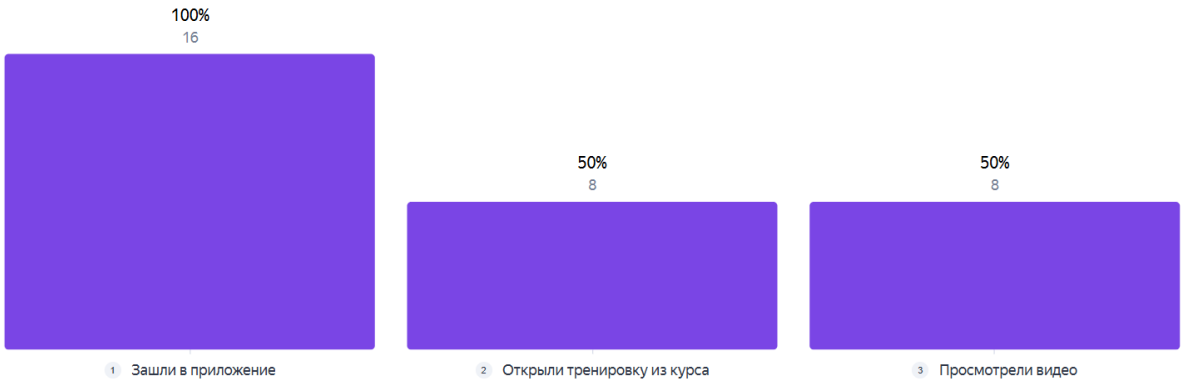


Рисунок 38 - Воронка прохождения тренировки от тренера

Рисунок 39 - Воронка поиска нового бесплатного курса

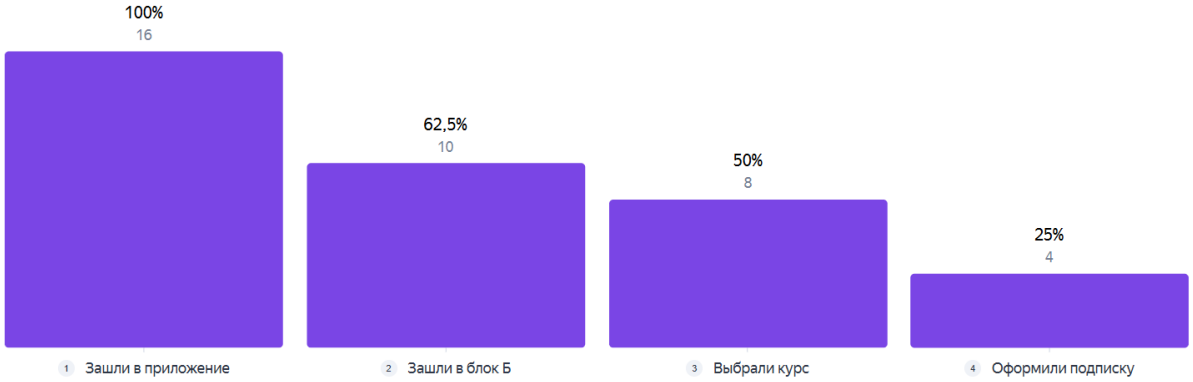


Рисунок 40 - Воронка поиска нового платного курса

Данные воронки провести базовую аналитику работы приложения, в будущем можно будет добавить больше сценариев, для лучшего понимания пользователя.

12 Тестирование приложения

12.1 Стратегия тестирования

Тестирование приложения «Trainova» проводилось с целью обеспечения качества и соответствия функциональным требованиям, описанным в техническом задании. Основные аспекты тестирования включали:

— функциональное тестирование – проверка работоспособности всех модулей приложения.

— UI-тестирование – оценка соответствия интерфейса макетам и требованиям юзабилити.

— интеграционное тестирование – проверка взаимодействия между клиентской и серверной частями.

— автоматизированное тестирование – использование инструментов Selenium для автоматизации тестов.

12.2 Основные модули для тестирования

12.2.1 Авторизация и регистрация

— регистрация новых пользователей.

— вход в систему с корректными и некорректными данными.

— восстановление пароля.

12.2.2 Главный экран

— отображение графика веса и активности.

— переход на другие экраны приложения.

12.2.3 Блок А (бесплатные тренировки)

— просмотр списка тренировок.

— фильтрация по группам мышц.

— прохождение тренировок (для авторизованных пользователей).

12.2.4 Блок Б (платные курсы)

— просмотр курсов.

— фильтрация по рейтингу и другим параметрам.

— покупка курсов и доступ к ним.

12.2.5 Личный кабинет

— управление профилем (смена пароля, почты, аватарки).

— просмотр истории платежей и подписок.

12.2.6 Функционал администратора

— создание и редактирование тренировок.

— управление упражнениями в пуле.

12.3 Результаты тестирования

12.3.1 Ручное тестирование

Были проведены тест-кейсы для всех ролей пользователей (неавторизованный, авторизованный, администратор). Основные результаты:

— все критичные функции (P1) работают корректно.

— ошибки валидации данных (например, некорректный email или пароль) обрабатываются правильно.

— навигация между экранами осуществляется без сбоев.

12.3.2 Автоматизированное тестирование

Для автоматизации тестирования использовался фреймворк Selenium. Были написаны тесты для:

— авторизации и регистрации.

— добавления тренировок с упражнениями (для администратора).

— навигации по основным экранам приложения.

— пример кода автоматизированного теста:

12.3.3 Выявленные проблемы и их устранение

В ходе тестирования были обнаружены следующие баги:

Backend-баги:

1. Баг отображения графика на карточке курса:

— описание: Некорректное распределение процентов загрузки мышц (сумма не равна 100%).

— исправление: Добавлена проверка на сумму процентов.

— статус: Исправлен.

2. Данные пользователя имеют неограниченную длину:

— описание: Поля имени, фамилии и описания не ограничены по длине.

— исправление: Введены ограничения на длину вводимых данных.

— статус: Исправлен.

Frontend-баги:

1. Ошибка форматирования описания тренера

— описание: Текст выходит за границы блока.

— исправление: Добавлены переносы текста и отступы.

— статус: Исправлен.

2. Некорректное переключение тренировок

— описание: Черный экран при быстром переключении упражнений.

— исправление: Оптимизирована загрузка контента.

— статус: Исправлен.

12.4 Заключение по тестированию

Тестирование подтвердило работоспособность основных функций приложения "Trainova". Все критические баги были устранены, что обеспечивает стабильную работу приложения для всех категорий пользователей. Автоматизированные тесты позволили ускорить процесс проверки и минимизировать риски появления критичных ошибок в будущем.

Заключение

В ходе выполнения данной работы были выполнены следующие основные шаги:

— разработано техническое задание, дизайн и брендбук приложения;

— спроектирована архитектура;

— разработаны backend и fronted;

— платформа была заполнена упражнениями и тренировками;

— привлечена тестовая аудитория;

— настроены метрики;

— приложение протестировано.

В результате было разработано приложение «Trainova» обладающие следующим функционалом:

— регистрация и вход;

— отслеживание своего прогресса;

— мотивирующие советы от нейросети;

— прохождение тренировок от приложения;

— создание своих курсов и тренировок;

— монетизация свои курсы;

— прохождение тренировок, созданных тренерами;

— комментирование тренировок;

— редактирование профиля;

— оформление подписку на платные и бесплатные курсы тренеров.

Список используемых источников

1. Что такое API и что о нём нужно знать веб-разработчику [Электронный ресурс]. – Режим доступа: https://practicum.yandex.ru/blog/chto-takoe-api/ – Заглавие с экрана. – (Дата обращения 29.05.2025).

2. Плюсы и минусы FastAPI в 2023 [Электронный ресурс]. – Режим доступа https://habr.com/ru/articles/748552/ – Заглавие с экрана. – (Дата обращения 30.05.2025).

3. СУБД PostgreSQL: почему её стоит выбрать для работы с данными [Электронный ресурс]. – Режим доступа https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql/ – Заглавие с экрана. – (Дата обращения 02.06.2025).

4. SimpleOne [Электронный ресурс]. – Режим доступа https://simpleone.ru/glossary/docker – Заглавие с экрана. – (Дата обращения 01.06.2025).

Приложение А

Все приложения также выложены на GitHub команды, где их легче разобрать.

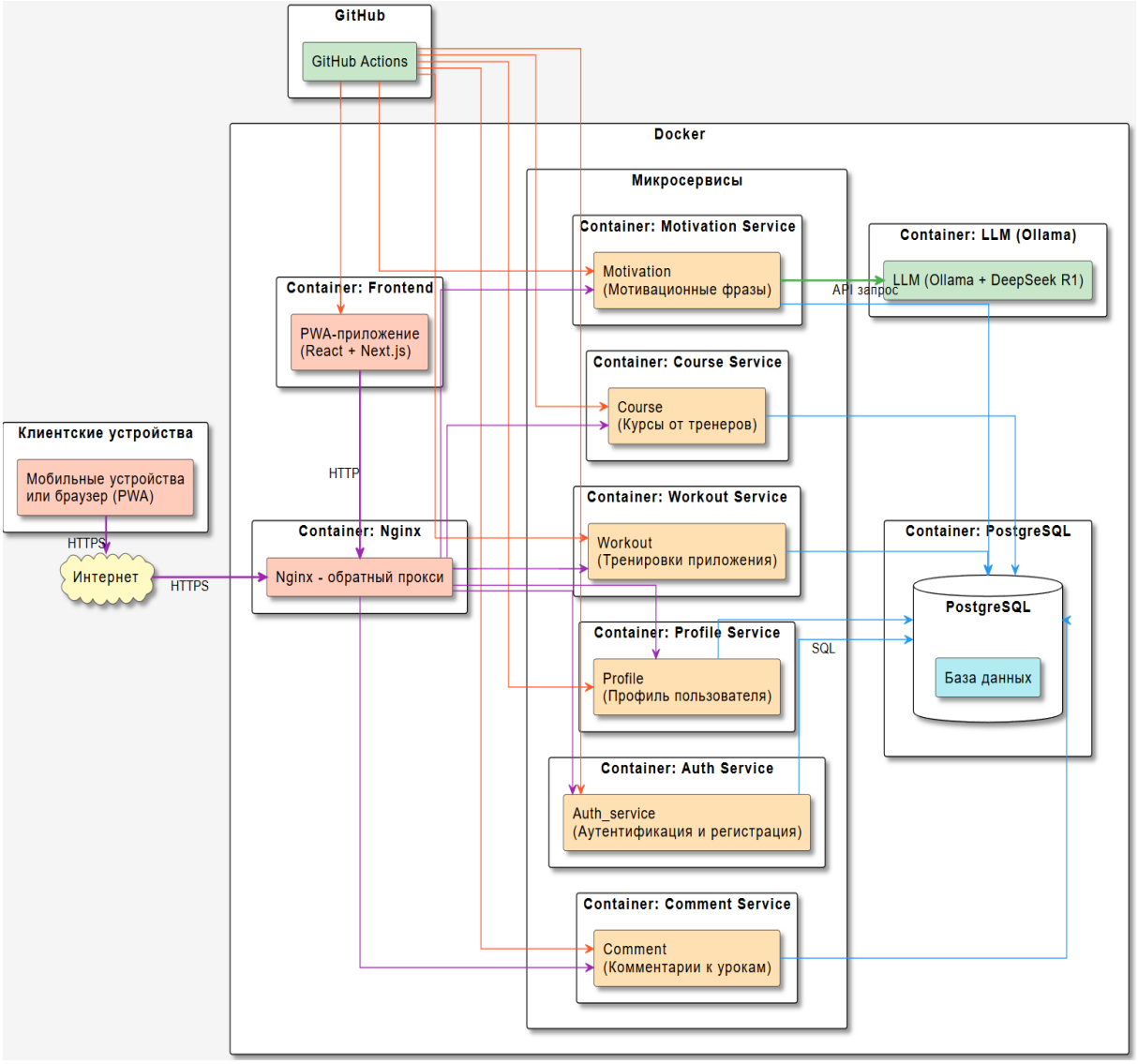


Рисунок 41 - Диаграмма развёртывания.

Приложение Б

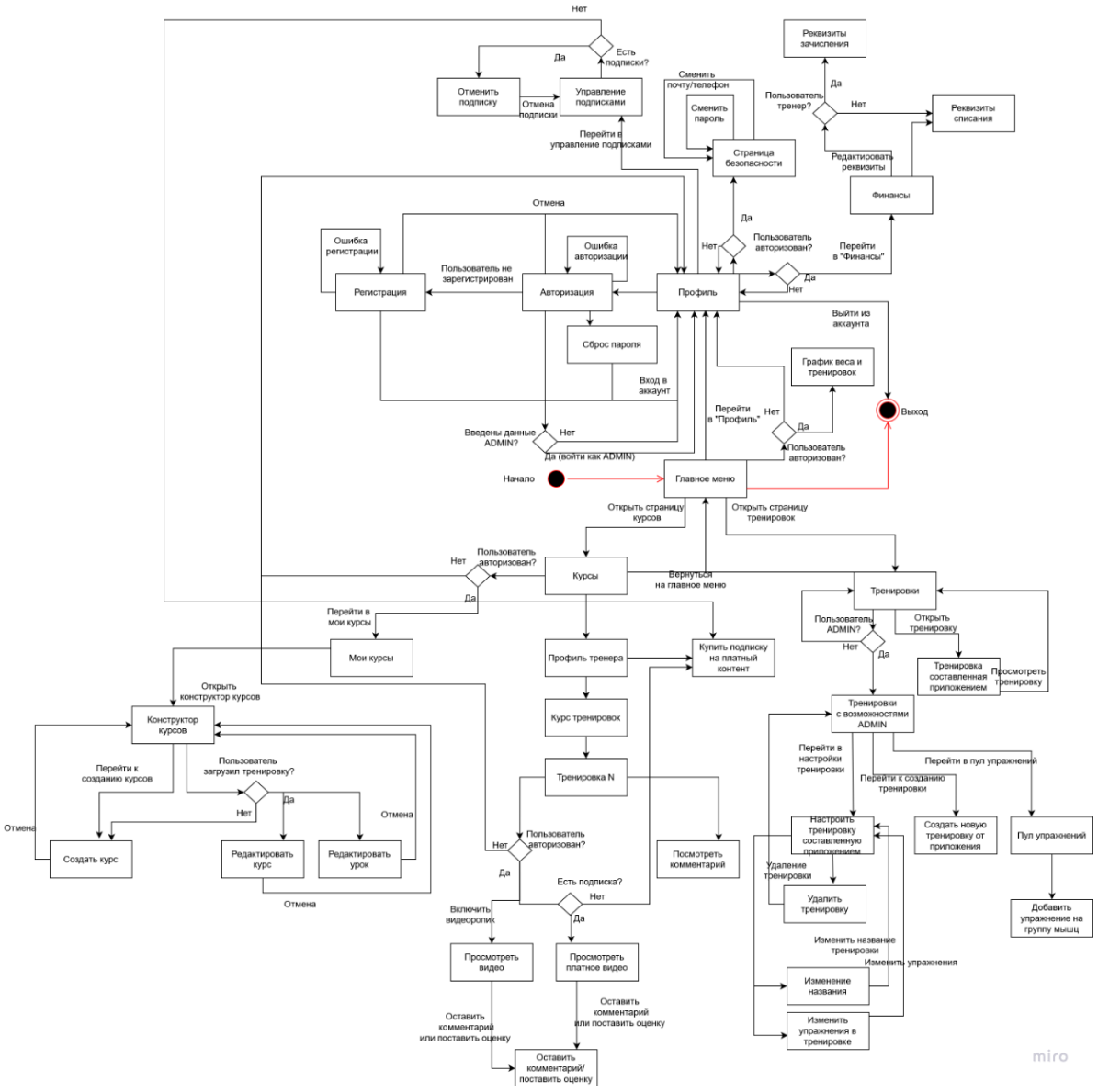


Рисунок 42 - Диаграмма состояний

Приложение В

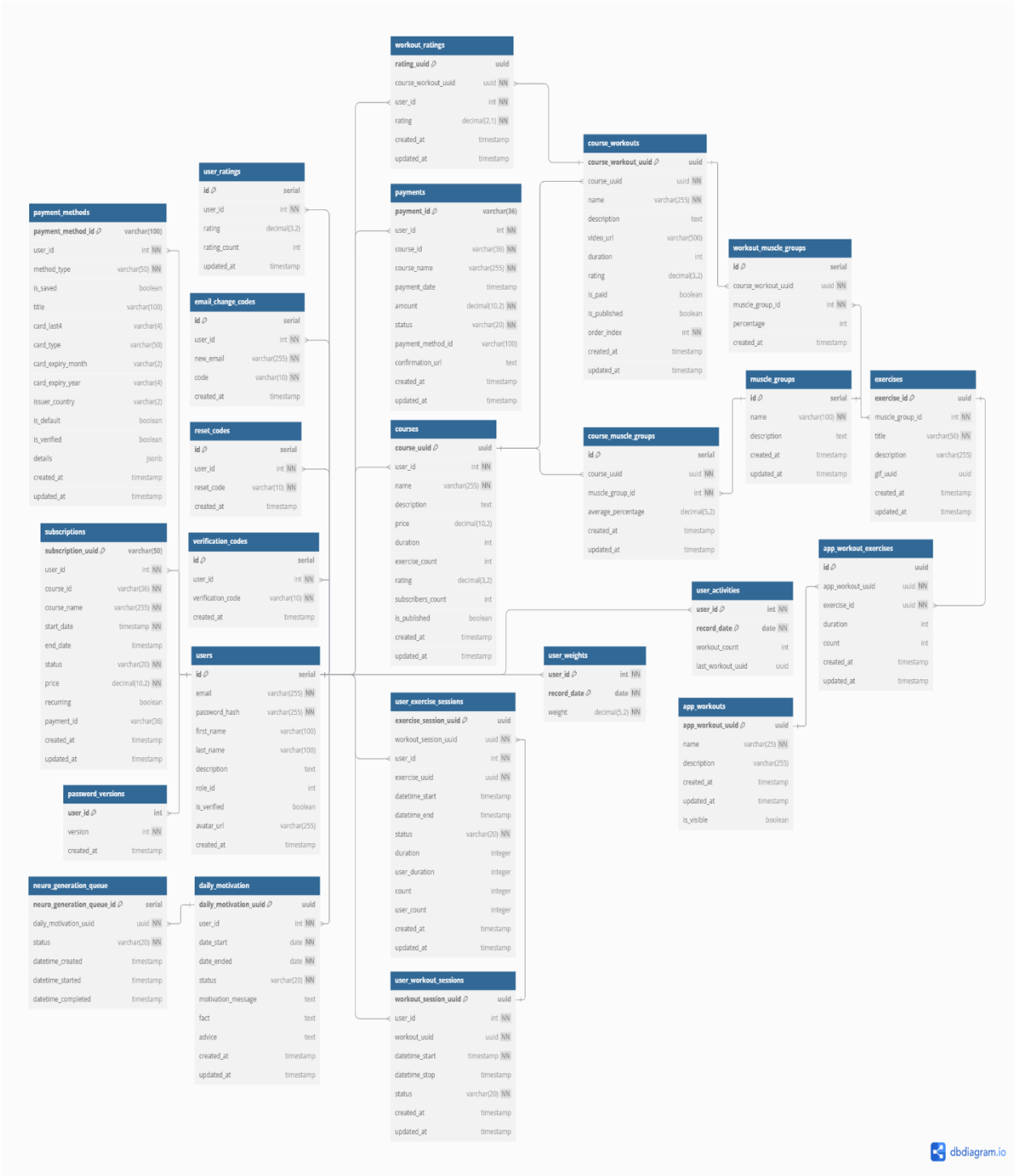
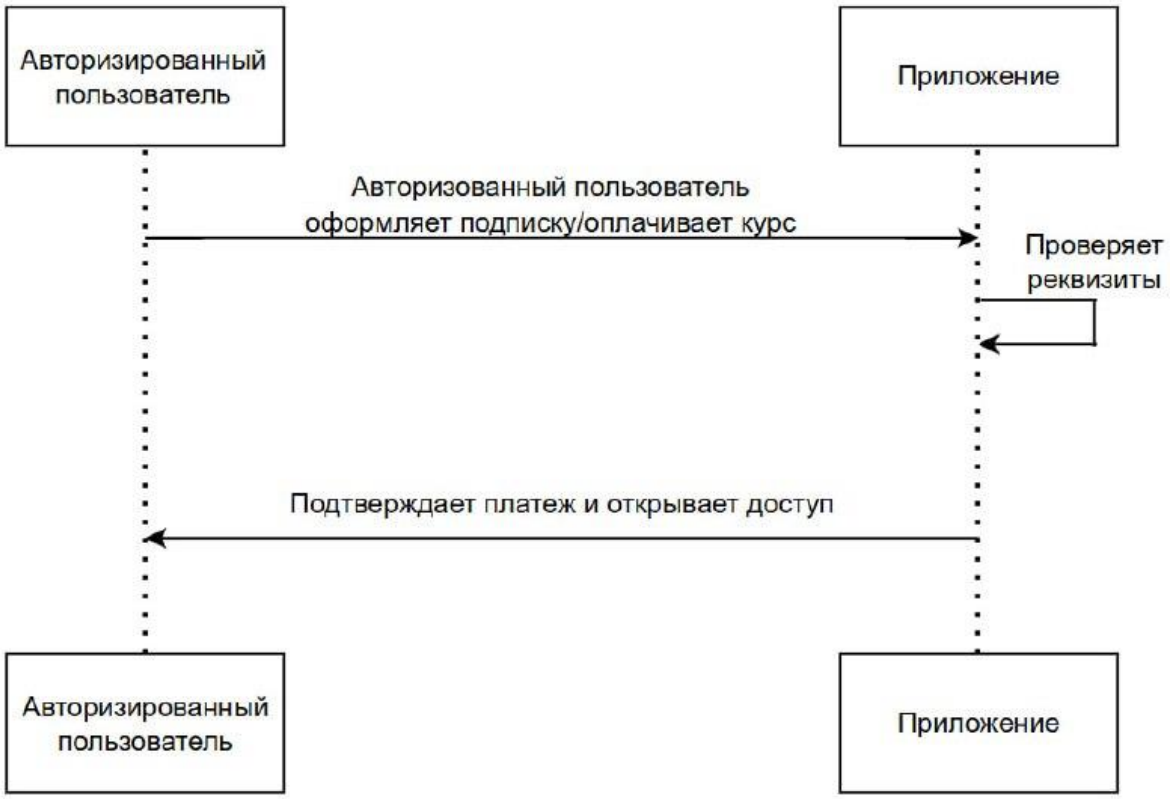
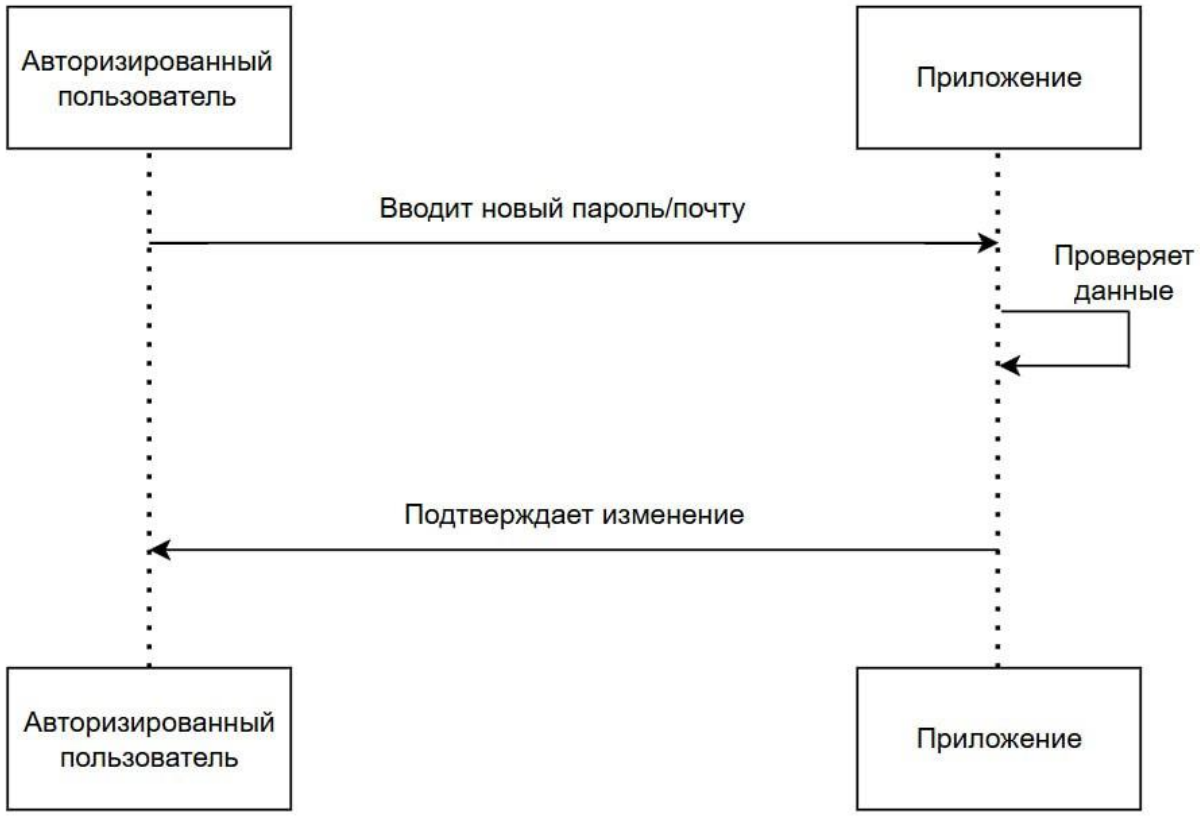


Рисунок 43 - ER диаграмма

Приложение Г


Авторизированный пользователь
Приложение
Авторизованный пользователь оформляет подписку/оплачивает курс
Проверяет реквизиты
Подтверждает платеж и открывает доступ
Авторизированный пользователь
Приложение


Авторизированный пользователь
Приложение
Вводит новый пароль/почту
Проверяет данные
Подтверждает изменение
Авторизированный пользователь
Приложение

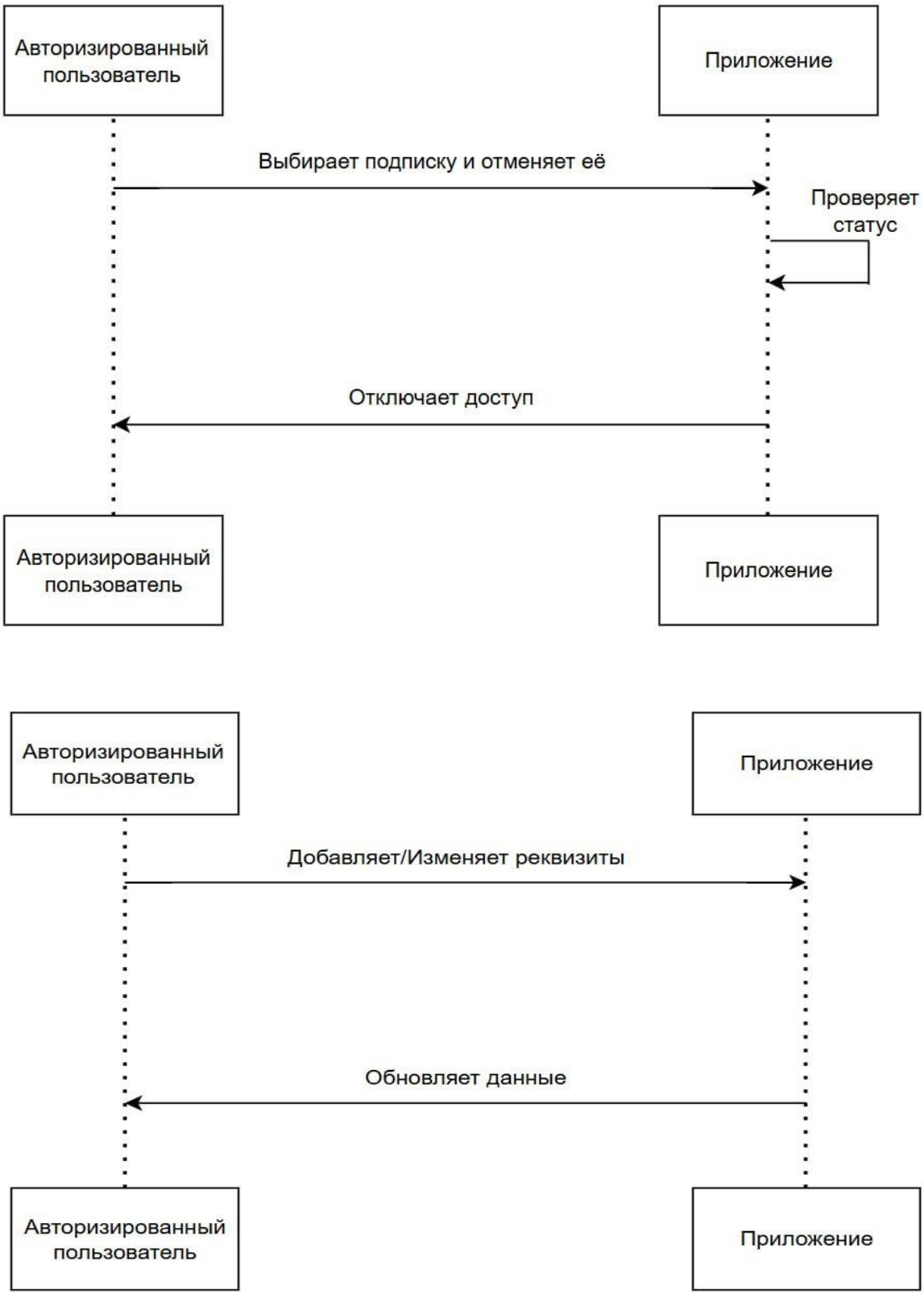


Рисунок 44 - Диаграмма последовательностей