Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

Факультет **Прикладной информатики**

Направление подготовки **09.03.03 Прикладная информатика**

Образовательная программа **Мобильные и сетевые технологии**

К У Р С О В О Й П Р О Е К Т

Тема: «Мобильное приложение для отслеживания поставленной цели»

Обучающийся: Блинова Полина Вячеславовна, К3139

Санкт-Петербург 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном быстро меняющемся мире саморазвитие и достижение личных целей становятся критически важными для успеха и благополучия. Актуальность курсовой работы обусловлена растущей потребностью в инструментах самосовершенствования в условиях современного динамичного мира. Рынок мобильных приложений для личностного роста демонстрирует значительный потенциал, однако многие существующие приложения имеют недостатки в плане эргономики интерфейса, отсутствия наглядной визуализации прогресса и ограниченных возможностей для обмена опытом между пользователями. Целевая аудитория проекта — пользователи Android, заинтересованные в отслеживании прогресса своих целей.

Целью курсовой работы стала разработка мобильного приложения для Android, позволяющее пользователям создавать ежедневные отчеты о прогрессе, просматривать отчеты других пользователей, а также визуализировать свой прогресс по достижению цели.

В задачи проекта входило формирование требований к функциональности, описание архитектурной схемы приложения, а также необходимых технических характеристик. Также было необходимо спроектировать структуры базы данных, реализовать API для просмотра отчетов, API для создания и удаления отчетов, прототип экрана просмотра отчетов с полным функционалом, прототип экрана создания отчетов, прототип экрана визуализации прогресса. В задачи проекта также входила реализация UI просмотра отчетов пользователей, UI создания отчетов, UI визуализации прогресса, а также подключение API для просмотра отчетов пользователей, API для создания отчетов пользователей и API для визуализации прогресса. Задачей проекта также являлось проведение функционального тестирования и исправление найденных ошибок в ходе тестирования.

1 Описание проекта

Проект представляет из себя мобильное приложение, которое служит дневником достижений и самоанализа, позволяющее пользователям, во-первых, фиксировать события: пользователь может создавать отчеты о своих действиях и успехах, связанных с личностным развитием, например, о прочтении книги или об успешном выполнении сложной задачи на работе и так далее. Во-вторых, давать оценку: каждому отчету можно присвоить оценку по 10-балльной шкале, что позволяет количественно измерить свой прогресс. В-третьих, отслеживать динамику: приложение позволяет просматривать отчеты в виде списка, что дает возможность отслеживать прогресс во времени. В-четвертых, визуализировать прогресс: график показывает динамику оценок по дням, что помогает пользователю наглядно увидеть свои успехи и падения, а также отслеживать тренды и закономерности своего развития. И в-пятых, просматривать прогресс других пользователей, что добавляет социальную составляющую проекта: добавление возможности просмотра отчетов других пользователей создает уникальную среду, где пользователи могут учиться друг у друга, мотивировать друг друга и вместе двигаться к самосовершенствованию. Приложение становится не только инструментом для самоанализа, но и источником вдохновения и поддержки на пути к личностному росту.

Для реализации проекта были использованы язык программирования Dart с фреймворком Flutter — для клиентской части приложения — и язык программирования Go — для серверной части. Выбор именно таких языков программирования был обусловлен тем, что Flutter — кроссплатформенный UI-фреймворк, позволяющий разрабатывать приложения, которые могут работать на шести платформах: Android, iOS, Linux, MacOS, Web и Windows, — именно благодаря этому разработчики экономят время и ресурсы: не нужно

создавать и поддерживать отдельные версии приложений для каждой платформы [1]. Go же позволяет создать высокопроизводительный, надежный и масштабируемый API, который будет эффективно взаимодействовать с мобильным приложением, обеспечивая стабильную и быструю работу сервиса.

При работе над проектом также использовался сервис Figma для разработки дизайна мобильного приложения и Material 3 — комплексный набор гайдлайнов, компонентов и инструментов для создания пользовательских интерфейсов.

Приложение организовано в трехуровневой архитектуре, обеспечивающей разделение ответственности и четкое взаимодействие между компонентами в соответствии с рисунком 1.

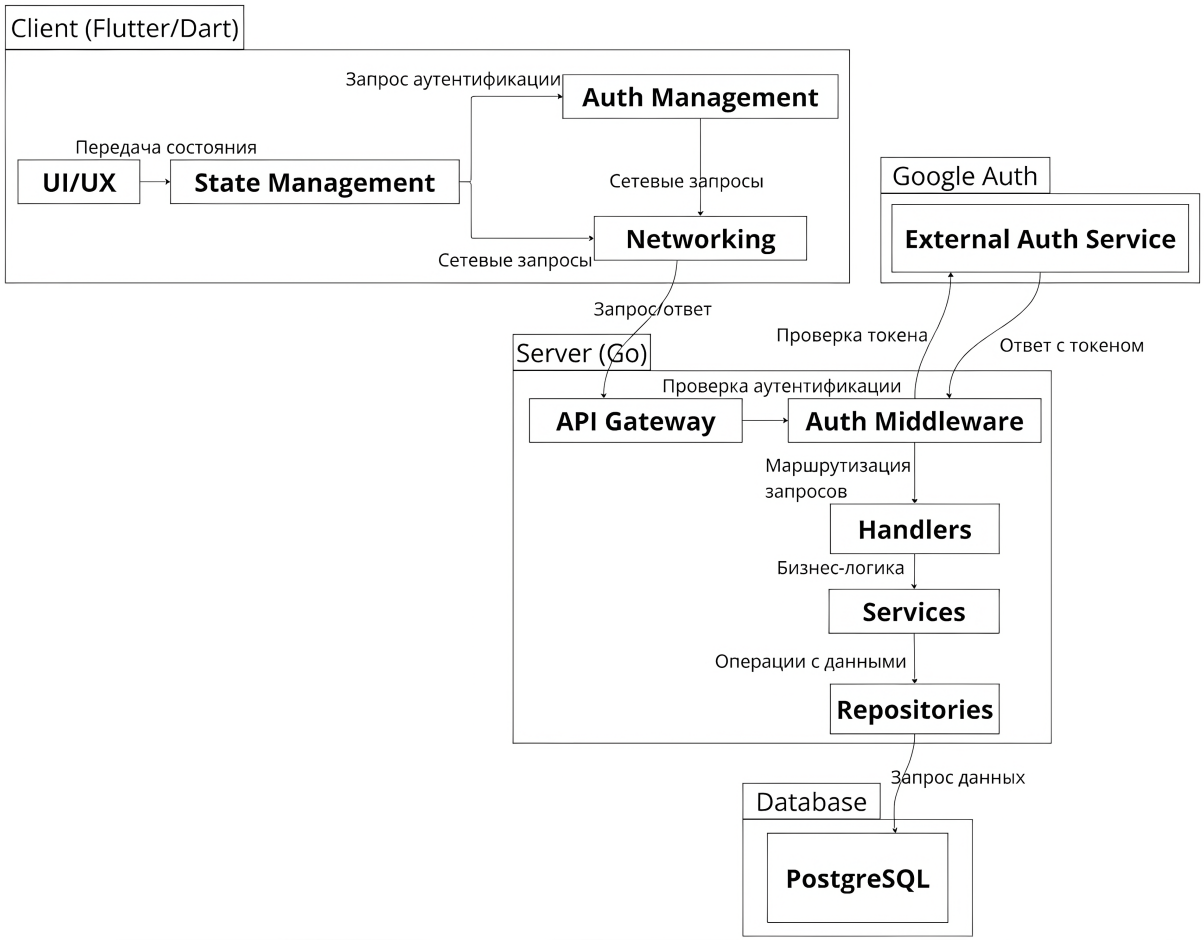

Рисунок 1 — Архитектура мобильного приложения

В самом низу находится база данных PostgreSQL, являющаяся основой для хранения всей информации, включая пользовательские данные и отчеты. Серверная часть, разработанная на Go, обрабатывает запросы и реализует бизнес-логику приложения. Начинается все с API шлюза, который выступает точкой входа для всех запросов от мобильного приложения. Для обеспечения безопасности используется промежуточное ПО аутентификации, которое проверяет подлинность пользователя. После аутентификации запросы направляются в соответствующие обработчики, которые, в свою очередь, вызывают методы сервисного слоя, где реализована основная бизнес-логика. Сервисы взаимодействуют с репозиториями для доступа к данным в базе данных. Клиентская часть, разработанная на Flutter, предоставляет пользовательский интерфейс и обеспечивает взаимодействие с пользователем. Она состоит из нескольких компонентов: UI/UX, отвечающего за создание видимых элементов, управления состоянием, которое управляет данными и обновляет интерфейс, управления аутентификацией, которое инициирует и обрабатывает процессы аутентификации. Для связи с сервером используется сетевое взаимодействие, а для аутентификации применяется внешний сервис аутентификации, который предоставляет токены. Данные передаются от клиента к серверу через API шлюз, где обрабатываются, а затем сохраняются или извлекаются из базы данных. Ответ возвращается в обратном порядке, достигая пользовательского интерфейса. Такая архитектура обеспечивает гибкость, надежность и возможность масштабирования приложения, позволяя легко добавлять новые функции и поддерживать его в будущем.

2 Работа над проектом

Перед началом работы над проектом командой был выделен ряд компонентов приложения. Были утверждены следующие этапы: подготовка и утверждение технического задания, проектирование и реализация серверной часть с API-эндпоинтами, проектирование пользовательского интерфейса и создание дизайна, реализация модуля просмотра отчетов, реализация модуля создания отчетов, реализация модуля визуализации прогресса, проведение комплексного тестирования приложения и представление и защита проекта.

На этапе подготовки командой были распределены задачи, необходимые для реализации проекта. Техническое задание в большей мере было составлено руководителем проекта, однако все участниками принимали активное участие в её доработке. Была четко сформулирована точная цель проекта, измеримая и достижимая.

Руководитель проекта, Алибеков Олег Олегович, предоставил всем участникам необходимый список материалов для изучения: например, разработчикам были предоставлены материалы по Flutter, а именно, изучение структуры проекта на Flutter [2], особенности языка Dart [3], основные виджеты [4], а также дополнительные материалы по виджетам, входящие в состав стандартных библиотек Flutter [5]. Руководитель проекта поставил определенный срок, до которого было необходимо все предоставленные материалы. Это позволило всем участникам проекта успешно выполнить свою работу.

На этапе создания пользовательского интерфейса был проведен анализ существующих аналогов. Изучив приложения конкурентов, было принято решение сделать мобильное приложение простым и удобным с акцентом на наглядную визуализацию прогресса.

На этапе создания серверной части приложения была разработана схема баз данных с информацией о пользователях, представленная в таблице 1.

Таблица 1 – Информация о пользователе

Параметр	Тип данных	Описание
Id	Serial primary key	Id пользователя
Email	Text Unique not null	Почта пользователя
Name	Text	Имя пользователя

Информация об отчетах представлена в таблице 2.

Таблица 2 — Информация об отчете

Параметр	Тип данных	Описание
Id	Serial primary key	Id отчета
Description	Text	Описание
Score	Int	Оценка отчета
Date	Timestamp	Дата
User_id	Int References Users(id)	Id пользователя

Во время этапа разработки модулей разработчики реализовали возможность добавления, просмотра отчета. Для визуализации прогресса была применена библиотека fl_chart, которая позволила отображать линейный график прогресса пользователя. Все экраны приложения были адаптированы для различных размеров устройств и оснащены возможностью прокрутки.

В рамках проекта были разработаны ключевые API-эндпоинты: GET-запрос для получения всех отчетов, что обеспечило интеграцию сервера с клиентской частью и доступ к данным; POST-запрос для создания новых отчетов с проверкой на сервере, ограничивающей длину текста до 500 символов; и DELETE-запрос для удаления отчетов по их идентификаторам.

Работа над проектом велась параллельно благодаря чёткому распределению ролей в команде. Каждый участник, завершив свои задачи, добавлял свой код в репозиторий на Github, обеспечивая непрерывную интеграцию. После объединения всего кода в закрытом репозитории было получено полностью функционирующее мобильное приложение.

На финальном этапе при составлении презентации для защиты участниками команды был выбран сайт Canva, так как это мощный онлайн-инструмент, который отлично подходит для создания презентаций благодаря своему удобному интерфейсу и широкому набору функций. Функция работы над презентацией в команде обеспечило совместный доступ для редактирования всем участникам проекта, что ускорило процесс подготовки к защите.

3 Мои задачи

В мои задачи входила реализация UI создания отчетов: реализовать ввод описания (текст до 500 символов), выбор оценки (по 10-балльной шкале) и автоматическую установку текущей даты (в формате UNIX), — подключение API для создания отчетов пользователей и проведение функционального тестирования.

Перед началом работы я внимательно изучила предоставленные руководителем материалы, так как это был мой первый опыт с языком программирования Dart и фреймворком Flutter. Первостепенной задачей было изучение синтаксиса языка Dart. Я старалась понять принципы работы этого языка, особенности объектно-ориентированного программирования, структуру классов и функций, а также правила написания кода, принятые в Dart. Это позволило мне в дальнейшем писать код, соответствующий стандартам, и более эффективно отлаживать его.

Параллельно с изучением Dart я приступила к исследованию структуры проектов на Flutter. Я изучала архитектуру приложений, организацию кода, взаимодействие различных компонентов, а также особенности построения пользовательских интерфейсов с использованием виджетов. Я не просто смотрела на примеры, а старалась понять, как они работают, как их можно настраивать и комбинировать для создания нужных мне элементов интерфейса.

Отдельное внимание я уделила изучению виджетов Flutter. Я изучала их разнообразие, назначение, способы настройки и использование. Я пробовала создавать собственные виджеты, чтобы лучше понять, как они взаимодействуют друг с другом и как их можно использовать для реализации конкретных задач.

Наконец, для начала реализации задач, я установила необходимую для разработки IDE (Integrated Development Environment), настроив все необходимые параметры и инструменты [6]. Это включило в себя установку плагинов для поддержки Dart и Flutter, создание необходимых конфигураций, а также ознакомление с возможностями IDE, такими как отладка кода и просмотр структуры проекта.

Моя роль в проекте была сосредоточена на разработке пользовательского интерфейса для создания отчетов, представленном на рисунке 2, что включало несколько ключевых аспектов. Прежде всего, необходимо было реализовать поле для ввода описания отчета. Это текстовое поле должно было обеспечивать возможность ввода произвольного текста, но при этом ограничивать длину введенных данных до 500 символов. Такая мера была необходима для соблюдения ограничений, установленных на стороне сервера и для обеспечения корректного хранения данных.

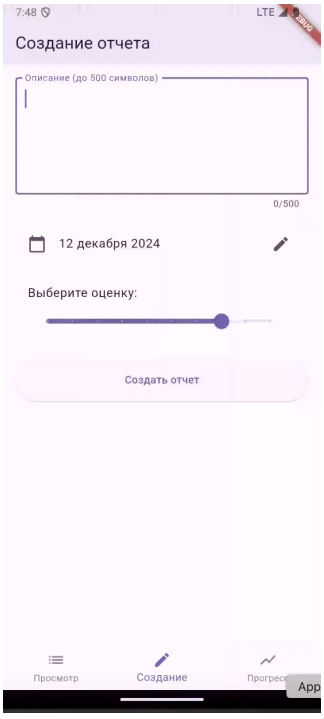

Рисунок 2 — Экран создания отчета

Следующим важным элементом интерфейса было поле для выбора оценки отчета. Этот компонент реализовывался в виде ползунка, который позволял пользователю выбрать оценку в диапазоне от 1 до 10.

Другим важным элементом пользовательского интерфейса была возможность выбора даты создания отчета. Вместо автоматической установки текущей даты, пользователю предоставлялась возможность вручную выбрать желаемую дату. При этом, выбранная пользователем дата преобразовывалась и сохранялась в формате UNIX timestamp. Это позволяло унифицировать формат хранения даты и времени на сервере, а также упростить дальнейшую обработку данных, давая пользователю большую гибкость при создании отчетов. Для реализации данной функции на клиентской стороне, для выбора даты пользователем, использовался виджет showDatePicker фреймворка Flutter. После выбора дата конвертировалась в UNIX timestamp. Для обработки даты на сервере был предусмотрен API-эндпоинт, принимающий timestamp в качестве параметра. Серверная часть, получив timestamp, верифицировала его. Важно отметить, что на сервере проводилась верификация выбранной даты, чтобы исключить возможность создания отчетов с датой в будущем, обеспечивая корректность и последовательность данных.

Помимо разработки пользовательского интерфейса, моя задача включала интеграцию с API для создания отчетов. Это включало отправку запросов на сервер с данными, полученными из UI, а также обработку ответа сервера. Я отвечала за отправку HTTP-запросов на сервер, используя метод POST, так как это соответствовало логике создания новых ресурсов. Данные, полученные из UI, а именно описание отчета (текстовое поле), оценка (числовое значение), и дата создания (в формате UNIX timestamp), формировались в JSON-объект. Этот JSON-объект отправлялся в теле запроса. При отправке запроса устанавливались необходимые заголовки, чтобы сервер

мог корректно интерпретировать полученные данные. После отправки запроса я обрабатывала ответ сервера. В случае успешного создания отчета я анализировала HTTP-код ответа (ожидая код 201, Created) и, в случае необходимости, обрабатывала полученные данные из тела ответа. При возникновении ошибок, например, если сервер возвращал код 400 (Bad Request) или 500 (Internal Server Error), я анализировала тело ответа для получения дополнительной информации об ошибке и отображала соответствующее сообщение пользователю.

Наконец, завершающим этапом моей работы было проведение функционального тестирования, реализованного UI и API. В ходе тестирования проверялась корректность работы всех элементов интерфейса, а также правильность взаимодействия с сервером. Особое внимание уделялось проверке валидации данных, обработке ошибок и обеспечению корректной передачи и сохранения данных. Тестирование проводилось с использованием различных тестовых данных и сценариев, для обеспечения надежной работы реализованного функционала. Так, бэкенд приложения подвергался нагрузочному тестированию с использованием более 10 000 отчетов для проверки его стабильности и производительности. На стороне мобильного приложения тестирование проводилось на устройствах с версиями Android выше 9, что включало проверку следующих функций: создание отчетов, их просмотр и визуализация прогресса. Особое внимание уделялось корректности работы ограничений, таких как максимальная длина текста, диапазон допустимых оценок и уникальность отчета для каждого дня. Для анализа времени загрузки и оптимизации производительности на Flutter, использовался инструмент профилирования Flutter DevTools.

Мне удалось работать планомерно, грамотно распределяя нагрузку благодаря грамотному руководству и четкому распределению задач со стороны руководителя.

За время работы над проектом я приобрела не только технические навыки, но и важные компетенции в командной работе. Во-первых, я существенно углубила свои знания в области разработки мобильных приложений на Flutter и языке Dart. Практическое применение полученных знаний позволило мне лучше понять принципы работы виджетов, управления состоянием приложения и взаимодействия с API. Я также научилась более эффективно использовать инструменты разработки, такие как IDE и Flutter DevTools, что позволило мне быстрее и качественнее писать код. Во-вторых, я освоила навыки командной работы, научилась эффективно взаимодействовать с коллегами, четко формулировать свои мысли и слушать других.

4 Взаимодействие с командой и руководителем

Вся работа над проектом, от обсуждения идей до координации задач, велась в мессенджере Telegram, который оказался крайне удобным инструментом для командной работы. Благодаря своей многофункциональности Telegram позволил оперативно обмениваться сообщениями, файлами и ссылками, а также создать групповой чат для обсуждения задач. Простота и скорость обмена информацией в Telegram способствовали поддержанию постоянной связи между участниками команды и позволяли быстро решать возникающие вопросы. Кроме того, для оперативного решения более сложных проблем, требующих личного обсуждения, команда использовала созвоны в Google Meet. Видеоконференции позволяли демонстрировать возникающие проблемы, быстро находить решения и обсуждать детали, что в итоге существенно повысило эффективность работы и позволило избежать задержек в реализации проекта.

Командная работа оказалась очень комфортной. Постоянное взаимодействие через созвоны и обсуждения в чате позволяло оперативно и эффективно решать возникшие в процессе реализации проекта вопросы. Активное участие каждого члена команды способствовало продуктивной реализации приложения. Возникающие затруднения в реализации RESTful API преодолевались за счет совместных усилий на созвонах с руководителем проекта, что подчеркнуло важность сплоченности коллектива. В целом, работа над курсовым проектом в команде стала для меня ценным опытом. Проект продемонстрировал, что совместная работа в команде облегчает работу и делает ее более эффективной.

Взаимодействие с моим руководителем, Алибековым Олегом Олеговичем, было предельно результативным, что сыграло ключевую роль в

моем развитии как Flutter-разработчика. На протяжении всей работы руководитель помогал с возникшими трудностями при разработке, оперативно координировал процесс командной работы и отвечал на все возникшие вопросы.

Я оцениваю работу моего руководителя, Алибекова Олега Олеговича, на высшую оценку. Он проявил себя как компетентный руководитель, четко определяющий временные рамки выполнения задач и эффективно организующий командную работу. Предоставленные руководителем учебные материалы в значительной степени способствовали освоению новых компетенций. Олег Олегович всегда был готов оказать помощь, давал ценные советы по разработке и содействовал в устранении возникающих ошибок.

ЗАКЛЮЧЕНИЕ

В ходе работы над проектом по разработке мобильного приложения в рамках курсовой работы была успешна достигнута цель — разработать мобильное приложение для Android, позволяющее пользователям создавать ежедневные отчеты о прогрессе, просматривать отчеты других пользователей, а также визуализировать свой прогресс по достижению цели. Команда смогла достигнуть поставленных задач и реализовать функционирующее мобильное приложение с сервером с базой данных и API-эндпоинтами для взаимодействий с клиентом, создала готовые дизайн-макеты и реализовала функциональные модули приложения.

Мой личный вклад в проект выразился в разработке функционального модуля создания отчета, где я приобрела навыки разработки мобильного приложения. По утвержденному техническому заданию я реализовала необходимый модуль, подключила API и провела функциональное тестирование приложения.

Таким образом, достигнутые результаты и выполнение поставленной цели демонстрируют успешную реализацию проекта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) https://education.yandex.ru/handbook/flutter/article/flutter-struktura-proekta

2) https://education.yandex.ru/handbook/flutter/article/flutter-struktura-proekta

3) https://education.yandex.ru/handbook/flutter/article/dart-osobennosti-yazyka

4) https://education.yandex.ru/handbook/flutter/article/widgets-basics-stless-stful-inherited

5) https://education.yandex.ru/handbook/flutter/article/widgets-standard-widgets

6) https://docs.flutter.dev/get-started/install

ПРИЛОЖЕНИЕ

Наименование проекта: мобильное приложение для отслеживания достижения целей личностного развития

Цель (назначение): разработать мобильное приложение для Android, позволяющее пользователям создавать ежедневные отчеты о прогрессе, просматривать отчеты других пользователей, а также визуализировать свой прогресс по достижению цели

Сроки выполнения: 01.11.2024 – 20.12.2024

Исполнитель проекта (руководитель проекта): Алибеков Олег Олегович

Термины и сокращения:

«Приложение» — Мобильное приложение для платформы Android, разрабатываемое с использованием фреймворка Flutter.

«Отчет» — Запись о прогрессе пользователя, содержащая описание действий за день, оценку достижения цели по 10-балльной шкале и дату.

«График прогресса» — Визуализация прогресса пользователя в виде графика, отображающая динамику достижения целей.

«API» — (Application Programming Interface) Интерфейс программирования приложений, позволяющий взаимодействовать между клиентской и серверной частями приложения.

«Flutter» — Фреймворк для разработки кроссплатформенных мобильных приложений на языке Dart.

«RESTful API» — Архитектурный стиль взаимодействия клиент-сервер, основанный на протоколе HTTP.

Технические требования (таблица 3):

Таблица 3 — Технические требования

Техническое требование	Язык разработки	Потребители

Платформа: Android, версия 9 и выше	Dart (Flutter).	Пользователи Android, заинтересованные в отслеживании прогресса своих целей
Архитектура: клиент-серверная с использованием RESTful API (Стандарты HTTP/1.1 для передачи данных, JSON для обмена данными.)	Dart (клиент), Go (сервер).	(Flutter и Backend)-разработчики
Создание отчетов: - пользователь может публиковать один отчет в день для каждой цели, - отчет включает: описание (текст, до 500 символов), оценку по 10-балльной шкале, дату (устанавливается автоматически).	Dart (Flutter для интерфейса), Go (сервер для обработки запросов).	Конечные пользователи
Просмотр отчетов: - возможность просматривать: свои отчеты и отчеты других	Dart (для отображения данных), Go (серверная логика).	Конечные пользователи

пользователей в одной ленте (Отчет включает: описание (текст, до 500 символов). оценку по 10-балльной шкале, дату (устанавливается автоматически).		
Визуализация прогресса (Линейный график (оценка, выставленная пользователем по достижению своей цели по дням), использование библиотеки для графиков (fl_chart).)	Dart	Конечные пользователи
Приложение должно загружаться не более чем за 3 секунды.	Dart (для клиента), Go (серверная оптимизация).	(Flutter и Backend)-разработчики, конечные пользователи.
Время отклика на действия пользователя — не более 1 секунды	Dart (для клиента), Go (серверная оптимизация)	(Flutter и Backend)-разработчики, конечные пользователи
Поддержка русского языка интерфейса (Использование пакетов Flutter	Dart	Конечные пользователи

(flutter_localizations для мультиязычности).		

Содержание работы представлено в таблице 4.

Таблица 4 — Содержание работы

Этапы работы	Сроки выполнения	Ответственный за этап
Сформировать требования к функциональности	01.11.2024 – 05.11.2024	Алибеков Олег Олегович
Описать архитектуру и технические характеристики	05.11.2024 – 08.11.2024	Алибеков Олег Олегович
Утвердить техническое задание	09.11.2024 – 10.11.2024	Алибеков Олег Олегович
Спроектировать структуру базы данных	11.11.2024 – 13.11.2024	Петрова Мария Валерьевна
Реализовать API для просмотра отчетов	13.11.2024 – 24.11.2024	Петрова Мария Валерьевна
Реализовать API для создания и удаления отчетов	20.11.2024 – 05.12.2024	Петрова Мария Валерьевна
Реализовать прототип экрана просмотра отчетов	11.11.2024 - 17.11.2024	Коновалова Кира Романовна
Реализовать прототип экрана создания отчетов	18.11.2024 - 24.11.2024	Коновалова Кира Романовна
Реализовать прототип экрана визуализации прогресса	25.11.2024 - 01.12.2024	Коновалова Кира Романовна

Реализовать UI просмотра отчетов пользователей	11.11.2024 - 24.11.2024	Гашимов Ильхам Фаррух оглы
Подключить API для просмотра отчетов пользователей	25.11.2024 – 08.12.2024	Гашимов Ильхам Фаррух оглы
Реализовать UI создания отчетов	01.11.2024 - 24.11.2024	Блинова Полина Вячеславовна
Подключить API для создания отчетов пользователей	25.11.2024 - 08.12.2024	Блинова Полина Вячеславовна
Реализовать UI визуализации прогресса	11.11.2024 - 24.11.2024	Сусликова Вероника Денисовна
Подключить API для визуализации прогресса	25.11.2024 - 08.12.2024	Сусликова Вероника Денисовна
Исправить найденные ошибки в ходе тестирования	11.12.2024 - 13.12.2024	Гашимов Ильхам Фаррух оглы
Подготовить презентацию проекта	14.12.2024 - 14.12.2024	Сусликова Вероника Денисовна

Основные результаты работы представлены в таблице 5

Таблица 5 — Результаты работы

Результаты работы (наименование)	Формы представления
Техническое задание	Утвержденное техническое задание, содержащее цели проекта, требования к функциональности и этапы разработки

Бэкенд с API	Рабочий сервер с настроенной базой данных и API-эндпоинтами для взаимодействия с клиентской частью
UI/UX Дизайн и прототипы	Готовые дизайн-макеты и прототипы экранов приложения, включая интерфейсы для создания отчетов, просмотра целей и визуализации прогресса
Функциональные модули приложения	Модуль просмотра отчетов. Модуль создания отчетов. Модуль визуализации прогресса (графическая визуализация прогресса пользователя на основе данных по 10-балльной шкале)
Презентация проекта	Подготовленная презентация проекта, включающая описание целей и способы достижения целей