**Федеральное государственное образовательное бюджетное учреждение высшего образования**

**«ФинансовЫЙ УНИВЕРСИТЕТ при Правительстве Российской Федерации»**

Факультет информационных технологий и анализа больших данных

**КУРСОВАЯ работа**

на тему:

«Мобильное приложение для управления проектами»

Руководитель:

Старший преподаватель

Милованов Д. М.

Москва 2022

Оглавление

[ВВЕДЕНИЕ 3](#_Toc104096532)

[ОСНОВНАЯ ЧАСТЬ 5](#_Toc104096533)

[Глава 1. Постановка задачи 5](#_Toc104096534)

[Глава 2. Описание предметной области 6](#_Toc104096535)

[Глава 3. Актуальность автоматизации 8](#_Toc104096536)

[Глава 4. Описание программы 9](#_Toc104096537)

[4.1. Алгоритмические решения 9](#_Toc104096538)

[4.2. Описание интерфейса программы 15](#_Toc104096539)

[4.3. Состав приложения 20](#_Toc104096540)

[ЗАКЛЮЧЕНИЕ 24](#_Toc104096541)

[СПИСОК ЛИТЕРАТУРЫ 25](#_Toc104096542)

# ВВЕДЕНИЕ

Зарождение управления проектами начинается в середине 20-го века. Первым проектом в привычном его понимании стала постройка ракетных систем “Поларис” в США. Проектом это стало из-за понимания, что на выходе должен получиться конкретный продукт, у которого жесткие ограничения по срокам ввиду гонки между США и СССР.

Тогда появилось понятие проекта как что-то, что делается какое-то время, имеет время старта и финала, а на выходе выдает уникальный продукт. В современном мире проекты могут быть самыми разными – от разработки сайтов до постройки зданий. Проектами можно назвать даже то, что ты делаешь самостоятельно для себя.

У проектов есть ключевые характеристики – ограниченность по времени и уникальный финальный продукт на выходе. Также проекты состоят из набора процессов и задач, которые не являются проектами по отдельности.

В настоящее время проекты зачастую настолько масштабны, что управлять ими вручную невозможно. Поэтому зародились методики управления проектами, но долгое время их внедрение было затруднительным для компаний, так как требовалась обработка большего количества данных с помощью ЭВМ.

В наше время для эффективного управления проектами необходимо использование программных средств – размеры проектов становятся масштабнее, становится тяжелее обрабатывать объем информации, расширяется количество сотрудников в компаниях, которые могут работать над одними и теми же проектами и задачами из любой точки мира

Требуемый функционал для такого программного обеспечения часто отличается и зависит от множества факторов. Например, от разброса сотрудников компании по стране и миру, от необходимости постоянного мониторинга дел зависит, нужна ли онлайн или офлайн платформа. Также важно в какой сфере работает компания. Для компаний из креативных сфер коллективное решение проблем и обсуждение идей могут стать приоритетной целью, тогда как в тяжелой промышленности зачастую уже есть четкий план, где нужно четкое распределение процессов проекта и контроль по выполнению задач.

Из всего этого вытекает актуальность создания нового ПО для удобного управления проектами. Поэтому мной была выбрана данная тема для курсовой работы – разработать мобильное приложение для управления проектами.

Объектом исследования является: ведение проектов по одной из существующих методологий при помощи мобильного телефона.

Задачи, по которым велась работа в ходе разработки данного приложения:

* Погрузиться в предметную область и изучить методологии управления проектами
* Проанализировать существующее программное обеспечение для управления проектами, в частности мобильные приложения
* Определить языка программирования и среды разработки
* Определить функционал приложения
* Реализовать алгоритм работы и интерфейс приложения на выбранном языке программирования
* Тестирование приложения

# ОСНОВНАЯ ЧАСТЬ

## Глава 1. Постановка задачи

В рамках курсовой работы требуется разработать кроссплатформенное приложение (Клиент) и приложение, которое занимается хранением и обработкой информации, (Сервер), выполняющее учет и отображение объектов предметной области, хранящихся в базе данных (БД), и предоставляющее справочную и статистическую информацию. Клиент должен быть с графическим интерфейсом.

Для реализации данного приложения был выбран фреймворк React Native на основе JavaScript, который позволяет создать кроссплатформенное нативное приложение с интерфейсом для IOS и Android. Как способ разработки был использован Expo – слой абстракций или набор инструментов, библиотек и сервисов для комфортной разработки. В Expo существуют инструменты для отладки и тестирования приложения. Нужно лишь установить на смартфон приложение Expo Go, подключиться с компьютером в одну локальную сеть, запустить Expo на смартфоне и открыть разрабатываемое приложение.

Далее было необходимо определить функционал будущего приложения. После изучения предметной области мной были выявлены следующие важные функции:

* Регистрация и авторизация пользователей;
* Разграничение функционала пользователей по ролям – администраторы и участники;
* Создание, редактирование и удаление данных;
* Добавление участников в проект;
* Создание, редактирование и удаление задач по проекту;
* Создание и удаление подзадач проекта;
* Фильтрация задач проекта по их статусу;
* Статистика по прогрессу подзадач;
* Визуализация данных о проектах, соответствующая выбранной мной методологии управления проектами Kanban.

## Глава 2. Описание предметной области

Управление проектами — это процессы по руководству работой команды для достижения целей проекта по заданным требований и ограничениям. Одним из наиболее распространенных программных инструментов управления проектами являются инструменты планирования. Они используются для установки последовательности действий проекта, назначения дат и ресурсов. Детализация и сложность плана, создаваемого инструментом планирования, варьируются в зависимости от используемой методологии управления проектами.

Методологией управления проектами является система принципов, правил и процессов. Самые популярные методологии отличаются друг от друга по структуре и требуют использования различных инструментов, а также требуют разработки программного обеспечения для управления проектами.

Методология управления проектами Agile – один из самых популярных методов управления проектами. Однако, Agile не является методологией сама по себе, это принцип управления проектами со своими целями и ценностями.

На основе Agile было создано несколько методологий со строгими правилами и инструментами планирования – Scrum, Kanban, Crystal, экстремальное программирование, Scrumban. Данные подходы четко сформулированы и имеют свои правила, что позволяет с их помощью сформировать четкую систему управления проектом и план для достижений целей компании.

Рассмотрим одну из самых популярных методологий в настоящее время – Kanban. Целью внедрения этой методологии является получение готового качественного продукта вовремя. Kanban начинается с визуализации, чтобы все процессы и задачи были на виду у команды. Для этого используется специальная доска и набор карточек с информацией о задачах.

Обязательным элементом данной методологии является доска задач, то как она должна примерно выглядеть показано на рисунке 1. К ней имеет доступ каждый участник проекта, чтобы всегда видеть на каком этапе находится задача. Для этого можно использовать как реальную, например пробковую, доску, так и виртуальную – популярным примером является программа Trello, в основе функционала которой как раз таки лежит методология Kanban.

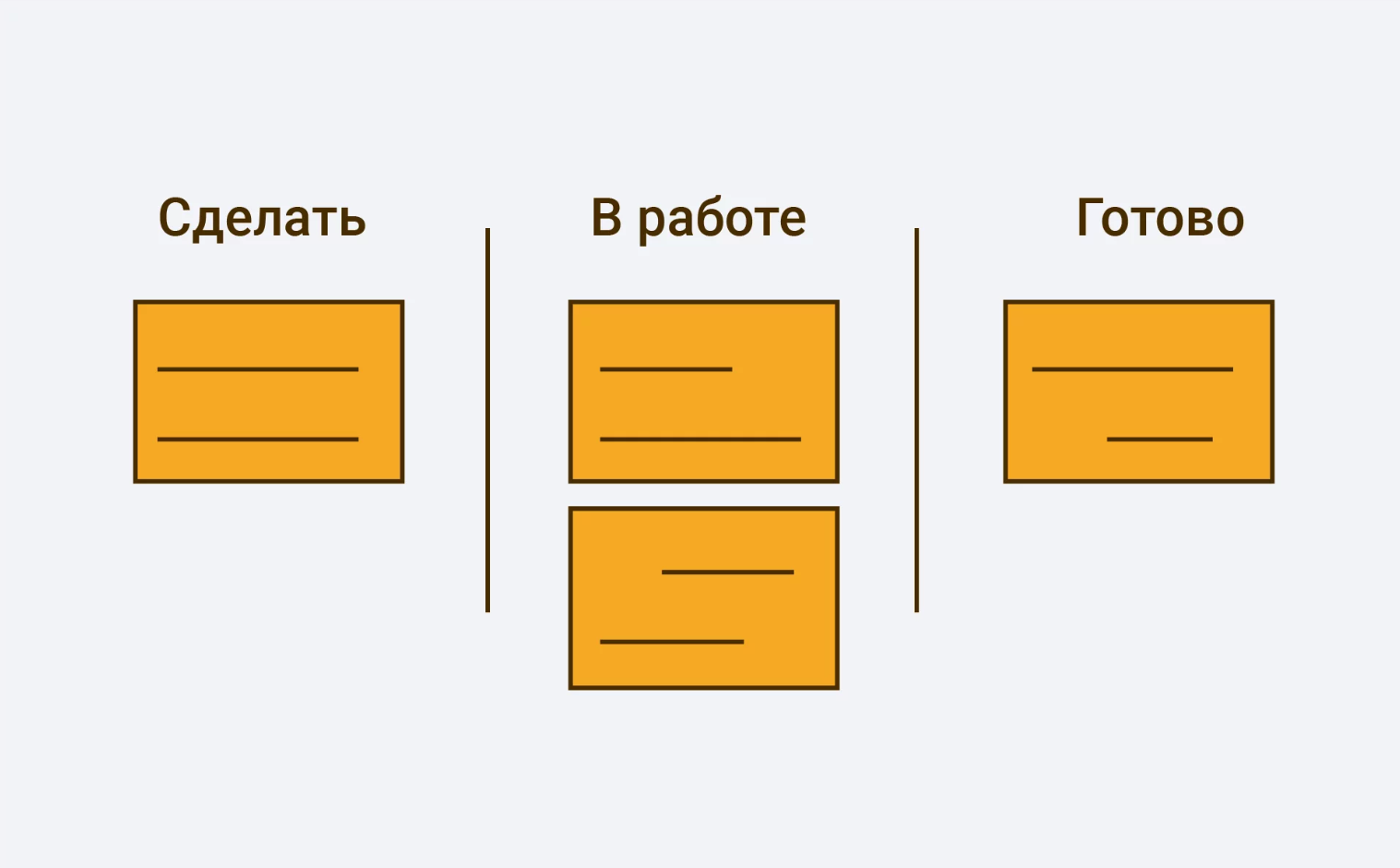


Рисунок 1 – Визуализация методологии Kanban

Имена столбцов могут меняться в зависимости от проекта и его требований, главное – сохранять их последовательность, которую в рамках данной методологии называют потоком.

Задачи проекта визуализируются с помощью Kanban-карточек, которые двигаются по потоку и перемещаются по столбцам в зависимости от их состояния.

Данная методология легла в основу разрабатываемого мной приложения. В нем представлен основной необходимый функционал Kanban-методологии – визуализация проектов и задач на виртуальной доске в виде потока Kanban-карточек.

## Глава 3. Актуальность автоматизации

В нынешнее время любой компании требуется наличие возможности быстро и легко адаптироваться к меняющимся условиям существования. Это является ключевым фактором для эффективного функционирования предприятия. Неотъемлемой частью существования каждой компании является управление процессами и проектами, а выбор правильной методологии управление проектом с учетом его требований и особенностей обеспечивает необходимую возможность к адаптации компании к

ПО для управления проектами может помочь в планировании, организации и управлении инструментами и ресурсами. В зависимости от сложности программного обеспечения, оно может управлять различными элементами управления. На сегодняшний день существует множество программных продуктов и услуг для управления проектами на базе ПК и браузеров, а также мобильных приложений, но управление проектами как наука постоянно развивается, появляются новые требования к проектам, создаются новые методологии управления, вследствие чего постоянно разрабатывается новое ПО, соответствующее современным тенденциям.

На рынке программного обеспечения для управления много гигантов таких, как Atlassian (имеет несколько таких ПО – Trello и Jira), Notion или Asana. В их функционал входит огромное количество возможностей для использования разных методологий управления проектами. Но даже они постоянно развиваются и совершенствуют свои приложения, чтобы соответствовать требованиям сегодняшнего дня.

Далее рассмотрим какая работа была проведена мной для реализации собственного ПО для управления проектами в виде мобильного приложения.

## Глава 4. Описание программы

### 4.1. Алгоритмические решения

Основная часть работы данного приложения заключается в использовании и хранении данных. Для этих целей было выбрано использование Firebase Authentication для регистрации и аутентификации пользователей, а также Cloud Firestore для хранения и использования данных пользователя и проектов.

Аутентификация Firebase – это функция аутентификации пользователей, которая предоставляется Firebase. Это система аутентификации на основе токенов, которая обеспечивает легкую интеграцию с большинством платформ.

Cloud Firestore — это NoSQL база данных, которая хранится в облаке и с которой iOS, Android и веб приложения работают напрямую с помощью нативных SDK. В Cloud Firestore используется синхронизация данных для обновления данных на каждом подключенном устройстве.

Модель данных Cloud Firestore поддерживает гибкие, иерархические структуры данных. Сохраняет данные в документах, которые в свою очередь хранятся в коллекциях. Документы могут иметь вложенные объекты и подколлекции.

Для подключения мобильного приложения к Firebase и последующего использования его инструментов, прежде всего, был создан проект в Firebase Cocnsole – Coursework-0-1, который представлен на рисунке 2.



Рисунок 2 – Курсовой проект на Firebase Console

Далее была проведена настройка необходимых инструментов, в нашем случае это Firebase Authentication и Cloud Firestore.

Для настройки аутентификации потребовалось только включить данный модуль для проекта и выбрать вариант аутентификации – мной был выбран способ Email/Password, использующий соответственно почту и пароль для регистрации и входа в систему. База данных, где хранится информация о зарегистрированных пользователей показана на рисунке 3.

После чего были использованы следующие функции из библиотеки ‘firebase/auth’:

* createUserWithEmailAndPassword() – функция, используемая для регистрации новых пользователей;
* signInWithEmailAndPassword() – функция для авторизации зарегистрированных пользователей;
* signOut() – функция выхода пользователя из системы.

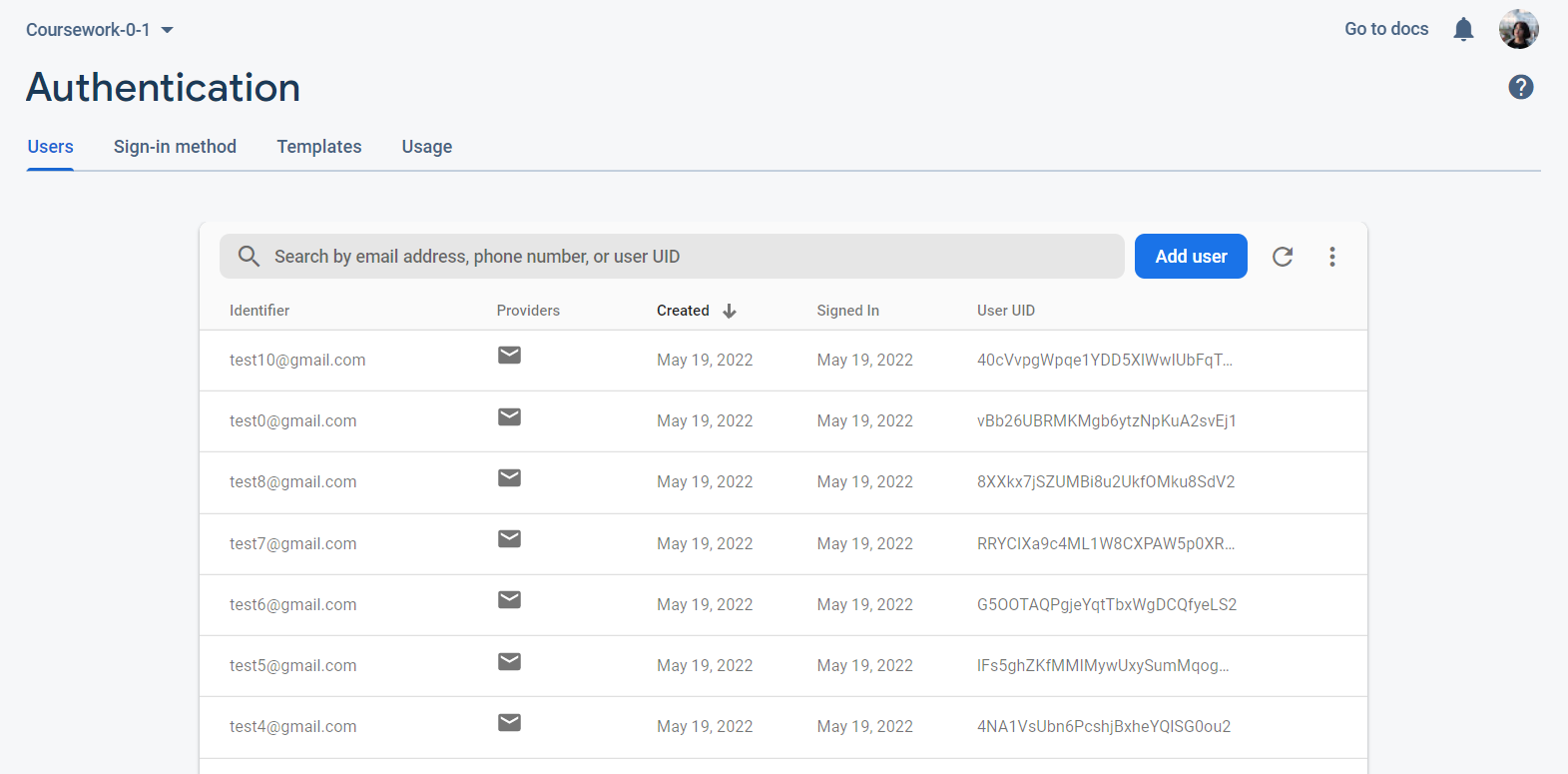


Рисунок 3 – Хранение данных о пользователях на Firebase

Для настройки хранения данных в Cloud Firestore, с помощью которых хранится вся информация о проектах и пользователях в приложении, потребовалось создать базу данных для проекта Coursework-0-1, она представлена на рисунке 4. Далее в данной базе данных были созданы необходимые коллекции данных:

* Users – данная коллекция необходима для хранения информации о пользователе, введенной на втором шаге регистрации. Хранятся имя, электронная почта и роль пользователя;
* ProjectsMain – в данной коллекции хранятся данные о проектах – имя проекта, даты старта и дедлайна, id участников проекта;
* ProjectTasks – здесь хранится вся информация о задачах и подзадачах проекта – название, описание, статус, дедлайн, также здесь хранится подколлекция tasks, в которой хранятся подзадачи.

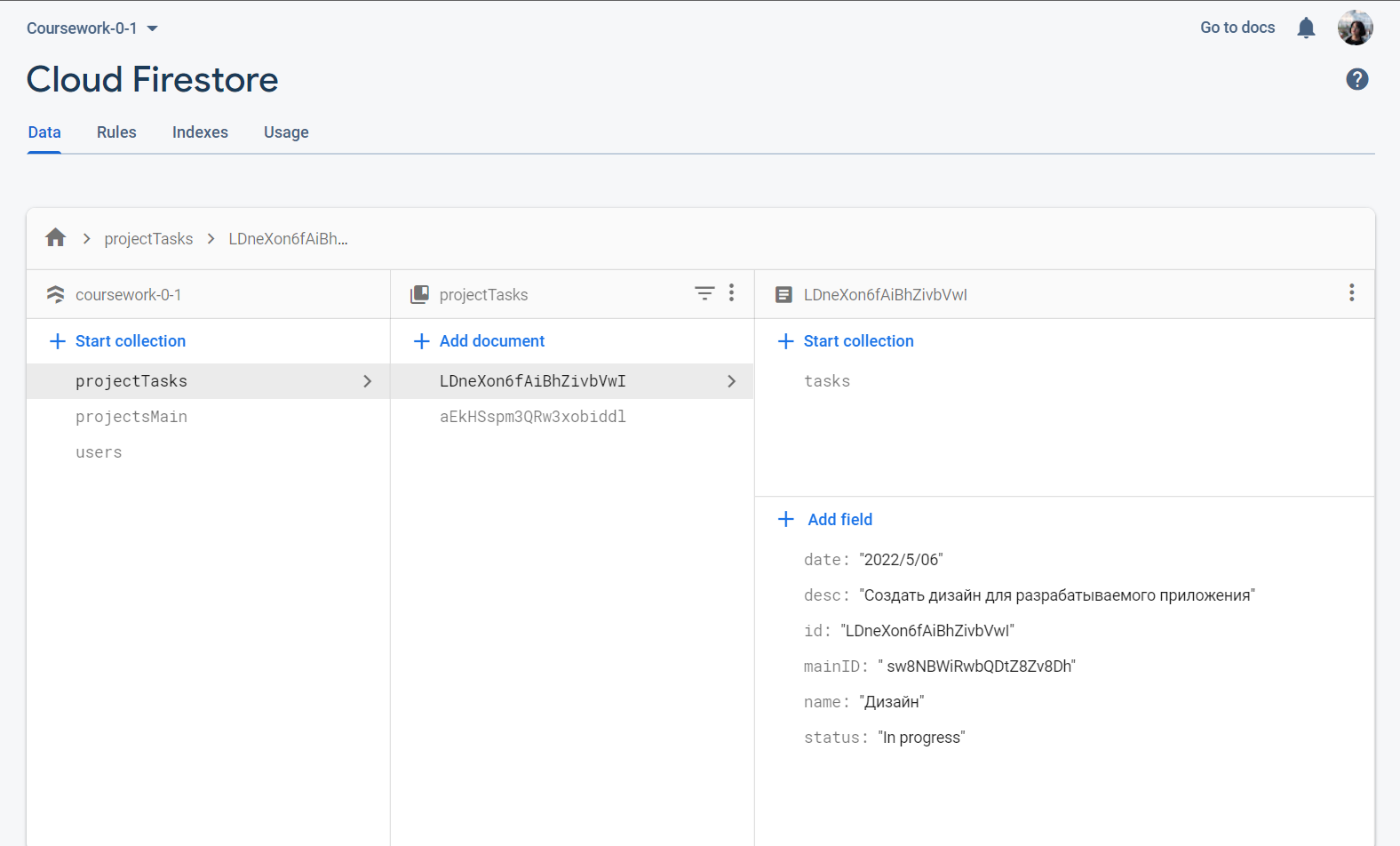


Рисунок 4 – База данных курсовой работы на Cloud Firestore

Чтобы работать с инструментами Firebase Console потребовалось подключить Firebase к мобильному приложению с помощью специального SDK, что показано на рисунке 5.

После настройки подключения приложения к Firebase осталось только написать необходимые функции для использования и отображения, хранения, создания, редактирования и удаления информации.

Все данные, отображаемые на страницах приложения, хранятся и используются с помощью функций Cloud Firestore – информация о пользователях, проектах, а также его задач и подзадач.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 – Интеграция Firebase SDK

Для записи и использования данных с помощью Cloud Firestore были использованы следующие функции, встроенные в библиотеку Firebase:

* addDoc() – функция, позволяющая создавать новые документы в выбранной коллекции с авто-генерирующимся ID документа. Используется при создании всех объектов в приложении – пользователей, проектов, задач и подзадач.
* collection(), doc() – функции, использованные для обращения к необходимым коллекциям и документам в базе данных проекта
* updateDoc(), arrayUnion(), arrayRemove() – функции обновления существующих документов и коллекций, использовалась для редактирования данных проектов, задач и подзадач
* deleteDoc() – функция удаления документов в выбранной коллекции, использовалась для удаления проектов, задач и подзадач
* getDoc(), getDocs() – функции чтения документов коллекции. Использовались для получения данных о проектах, задачах и подзадачах для последующей их визуализации в приложении
* query(), where() – функции запросов к документу. Использовались для получения данных о пользователях, проекте, задачах и подзадачах, удовлетворяющих конкретным условиям.

Для визуализации полученных данных было принято решение использовать компонент React Native Flatlist – это простой способ создать список данных с прокруткой. С помощью него представлено большинство данных – карточка пользователя в учетной записи, список с карточками проектов, задач и подзадач. Для этого используется два основных реквизита – массив данных, изначально полученный с помощью функций библиотеки Firebase, а также рендерируемые элементы данного массива данных, после чего отображаются нужные нам компоненты.

С помощью данного компонента реализован интерфейс для страниц со списком проектов, списком задач, подробной информации о задачах, а также для карточки пользователя на странице учетной записи.

Для форм создания, добавления или редактирования данных об элементах приложения была использована библиотека React Native Formik, с помощью функций которой данные введенные пользователем обрабатывались и перенаправлялись в функции создания, добавления и редактирования, основанные на функциях Firestore, таких как addDoc() и updateDoc().

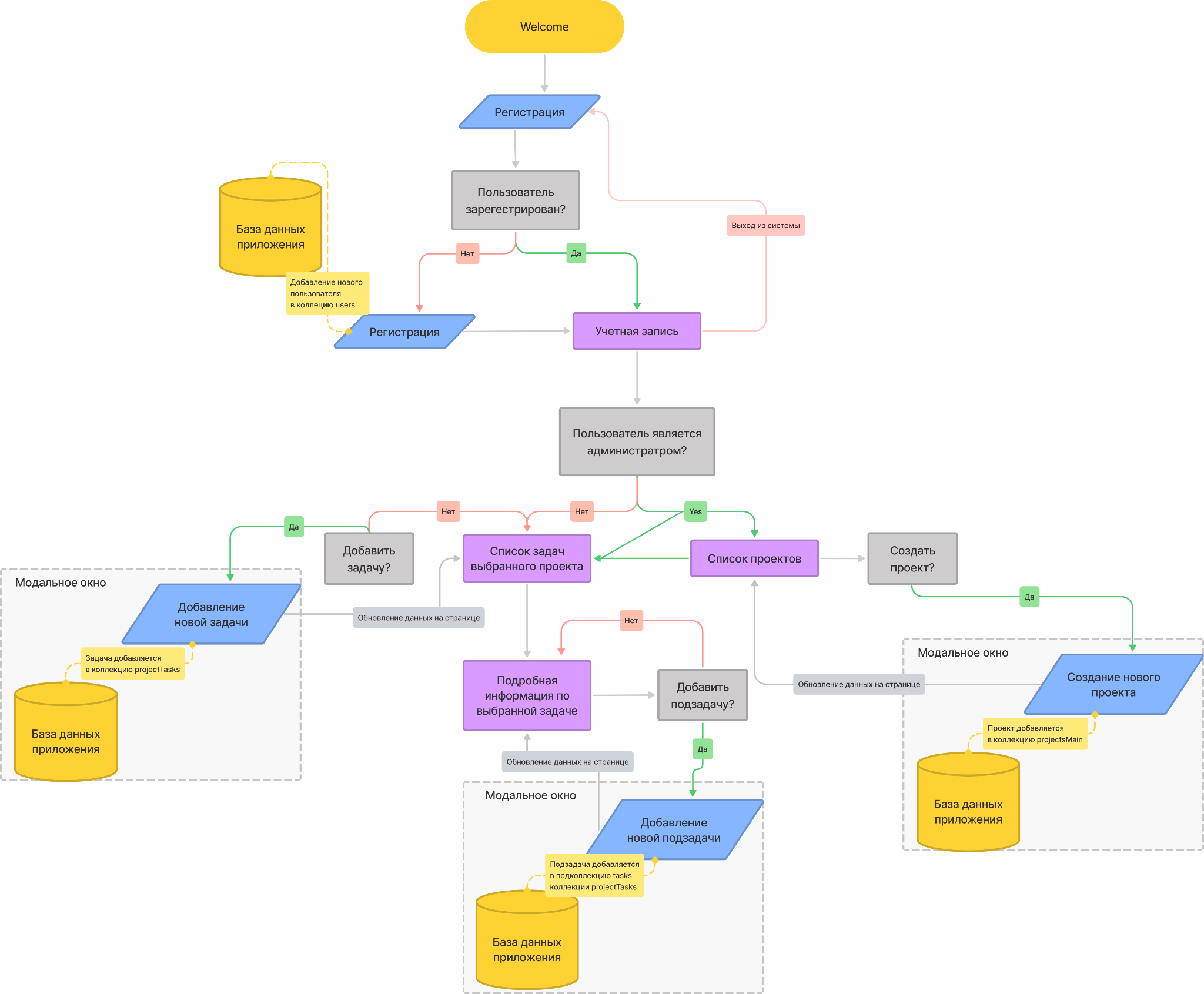


Рисунок 6 – Схема работы приложения

На схеме работы итогового приложения, которая представлена на рисунке 6, можно увидеть на каких этапах используется хранилище Cloud Firestore для записи пользователем новых данных и где используются и визуализируются полученные из хранилища данные.

### 4.2. Описание интерфейса программы

При запуске приложения пользователь попадает на Welcome экран, с которого он может перейти к регистрации/аутентификации. Пользовательский путь с этого экрана представлен на рисунке 7.

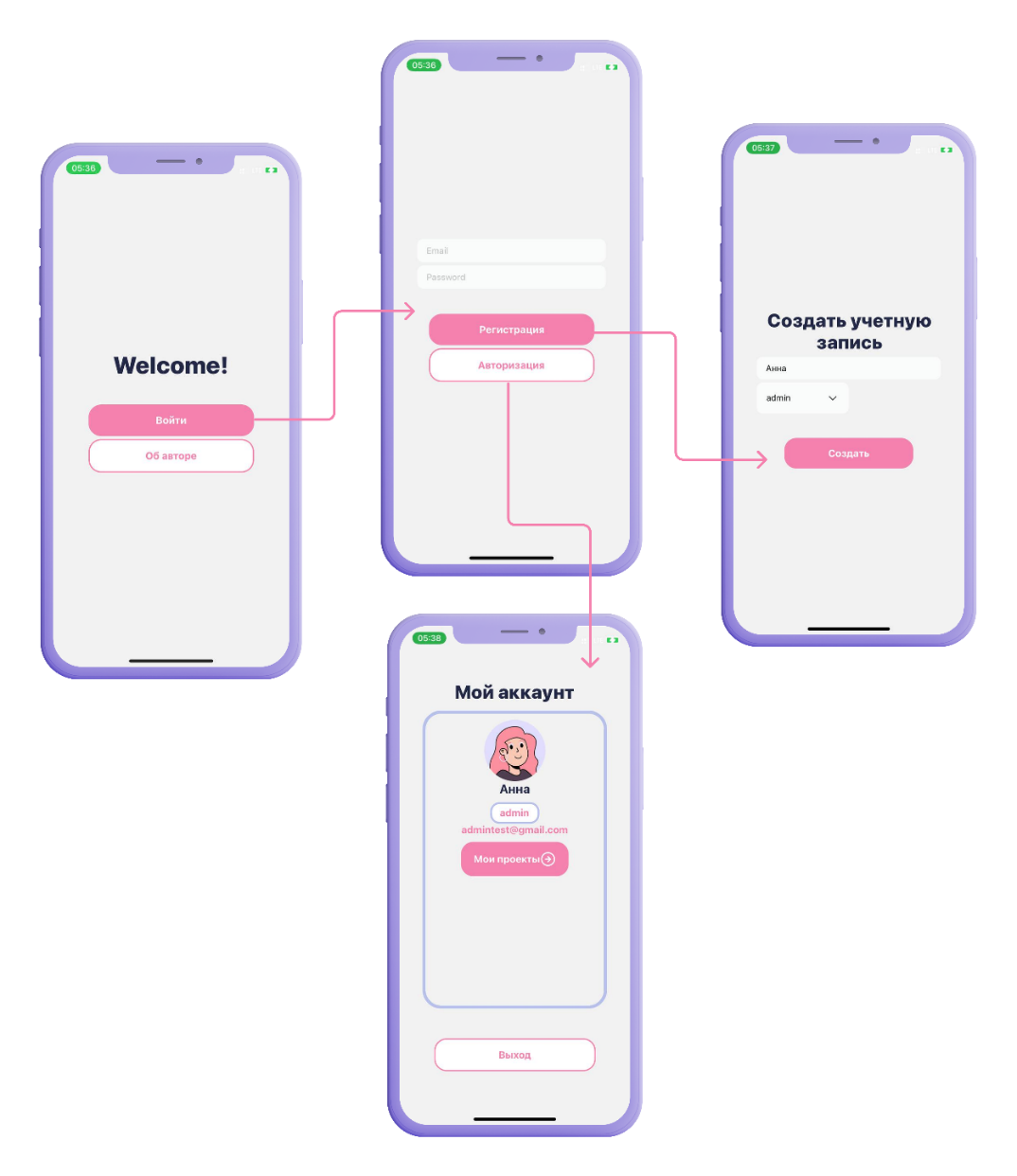


Рисунок 7 – Интерфейс приложения

При переходе на экран регистрации/аутентификации пользователю предлагается ввести электронную почту и пароль. При нажатии на кнопку регистрации пользователь направляется на экран следующего этапа регистрации, где ему потребуется ввести дополнительную информацию о себе, после чего он будет направлен на страницу учетной записи пользователя, как и если бы пользователь нажал на кнопку авторизации.

На странице учетной записи представлена основная информация о пользователе – имя, почта и его роль. Также на данной странице показан список проектов, в которых участвует пользователь.

Здесь же пользователю доступна кнопка, по нажатию на которую можно выполнить выход из системы, после чего произойдет перенаправление на страницу регистрации/авторизации.

Далее рассмотрим два сценария пользовательского пути в зависимости от его роли – администратор или участник.

1. Сценарий пользовательского пути для администратора:

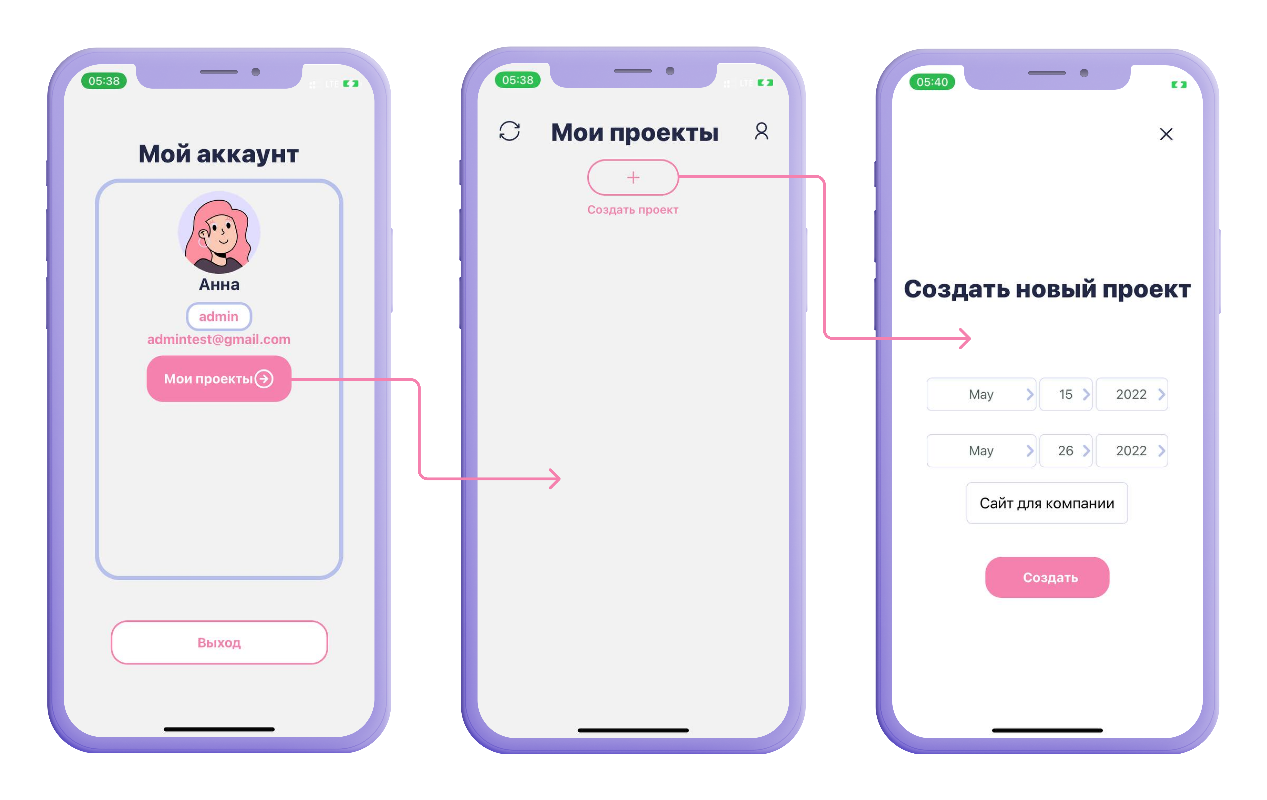


Рисунок 8 – Создание нового проекта

Если авторизованный пользователь является администратором, то он может перейти к странице проектов с расширенным функционалом – возможностью создавать новые проекты, редактировать и удалять существующие, что представлено на рисунке 8. С данной страницы администратор может перейти на страницу каждого проекта с задачами, которые он также может редактировать, удалять и создавать новые задачи, как показано на рисунке 9.

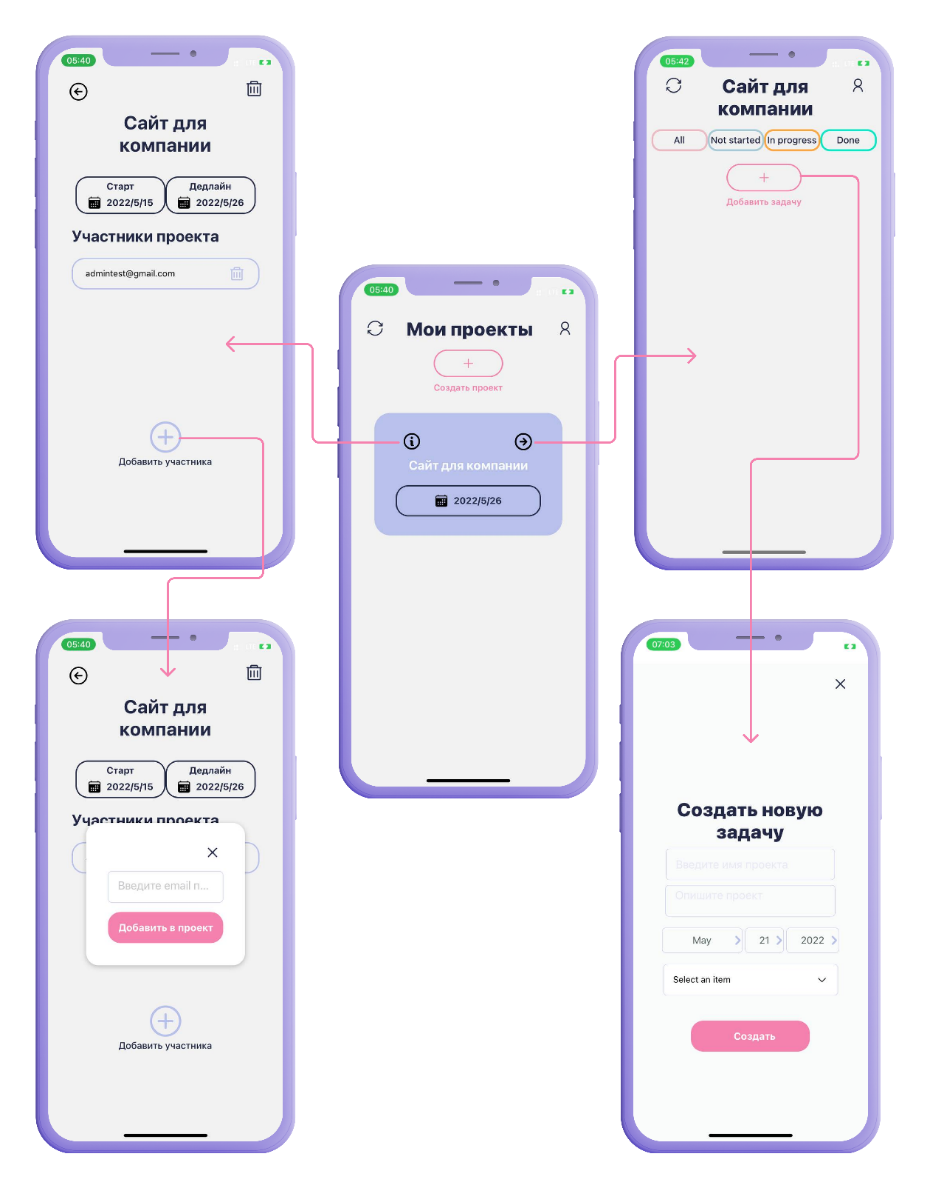


Рисунок 9 – Путь администратора со страницы проектов

На странице со списком задач проекта представлен функционал, отвечающий требованиям выбранной методологии Kanban – данный список визуализирует доску с Kanban-карточками, которые фильтруются по статусу. Для визуализации потока использован бар со статусами – при переходе между которыми отображаются только те карточки, которые соответствуют выбранному статусу. Данный интерфейс показан на рисунке 10.

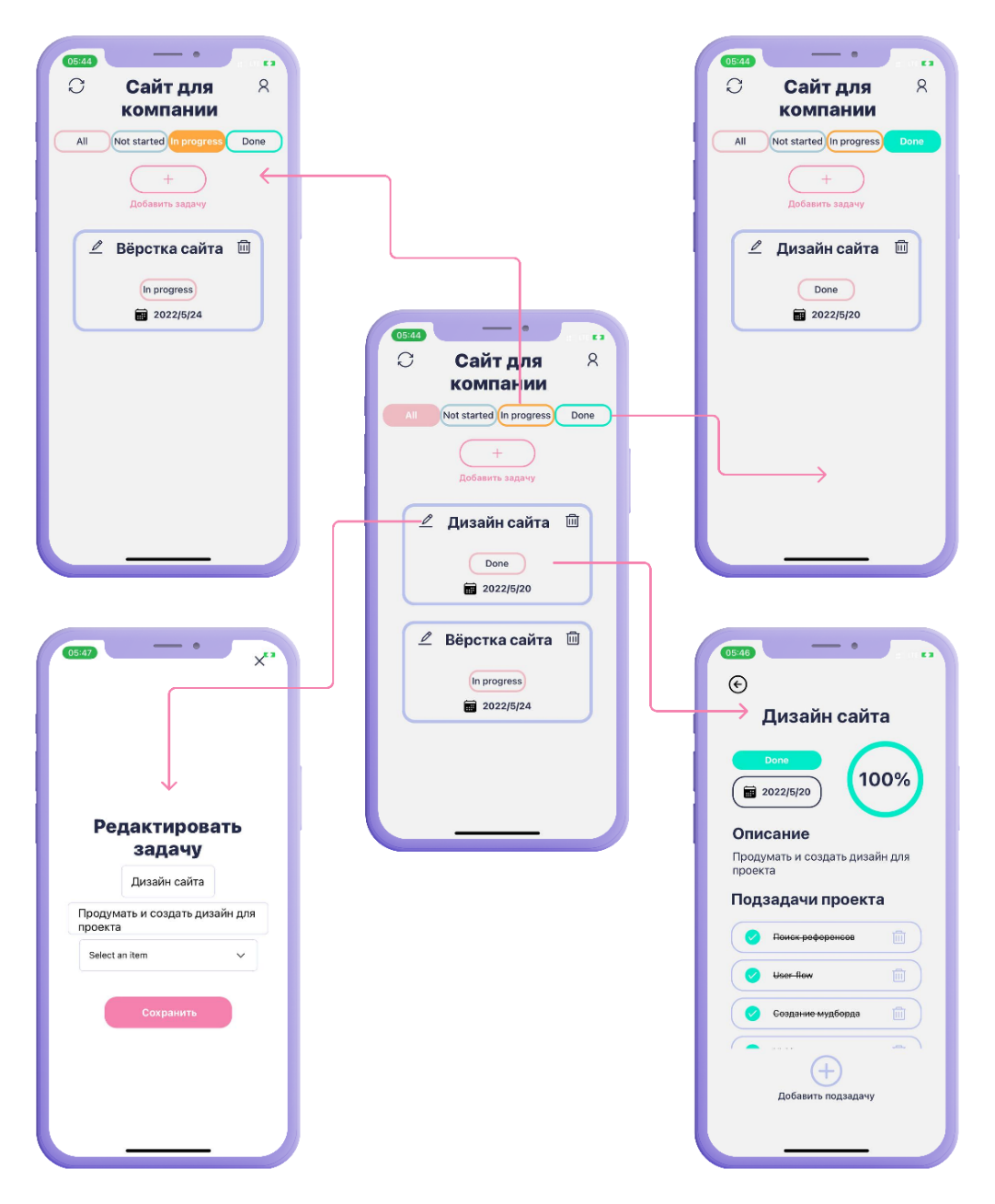


Рисунок 10 – Путь администратора со страницы задач

При нажатии на каждую задачу пользователь направляется на страницу с подробной информацией о данной задаче, а также списком подзадач, которые любой пользователь может создавать, удалять и отмечать их выполнение.

1. Сценарий пользовательского пути для участника

Если же текущий пользователь является участником, то на странице учетной записи для него доступен только просмотр списка проектов, в которых он участвует. По нажатию на проект он сможет перейти на страницу данного проекта, где представлены задачи по проекту, что представлено на рисунке 11. Участнику не доступны функции создания, редактирования и удаления проектов и задач.

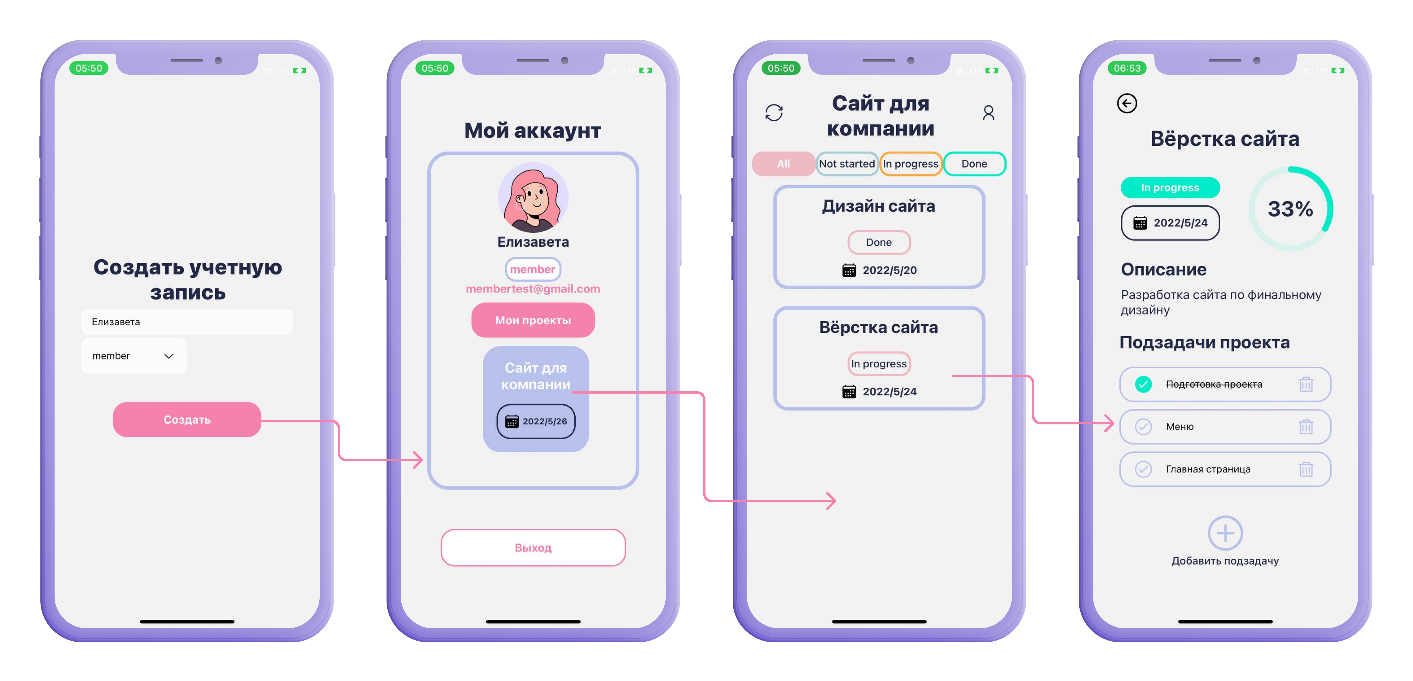


Рисунок 11 – Пользовательский путь участника проекта

При нажатии на задачу любой пользователь попадает на страницу подробной информации о данной задаче – название, статус, дедлайн, описание. Также на этой странице показаны подзадачи проекта и круговая диаграмма, показывающая прогресс по выполненным подзадачам. В независимости от роли пользователь может создавать новые подзадачи и отмечать их выполнение. Если пользователь отмечает подзадачу как выполненную, то меняется у нее меняется состояние иконки чек-бокса, а также текст подзадачи зачеркивается.

Главной задачей было реализовать функционал, отвечающий за методологию Kanban – было принято решение представить страницу со списком задач по проекту в виде доски с верхним меню фильтрации по статусу задачи. В данном приложении у задач есть 3 типа статуса: not started, in progress, done. Пользователь может отфильтровать отображаемый на странице список задач по нажатию на кнопку, соответствующей нужному статусу. Также был добавлен пункт All, чтобы у пользователя была возможность просмотреть все задачи по проекту. С точки зрения UX/UI -дизайна была добавлена функция изменения состояния выбранного пункта меню. Реализация данных функций Kanban-доски представлена на рисунке 12.

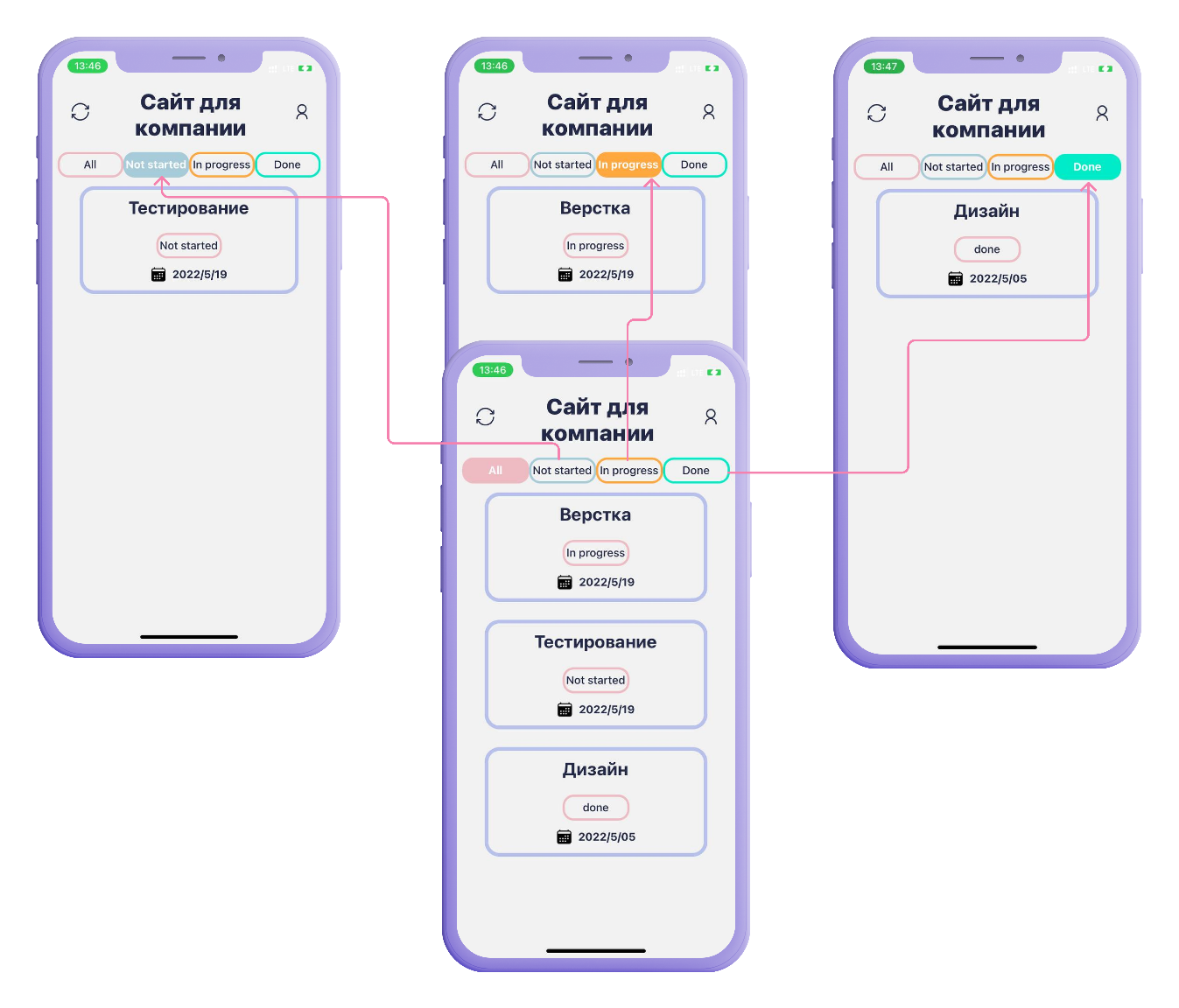


Рисунок 12 - Реализация методологии Kanban

Реализация доски Kanban в приложении позволяет не только визуализировать отслеживание по статусу благодаря фильтрации, но также и перемещать задачу по мере изменения её статуса – для этого администратору нужно изменить статус задачи, что доступно ему в форме изменения информации о задаче при нажатии на иконку редактирования.

### 4.3. Состав приложения

Рассмотрим файловую структуру приложения

Запуск приложения начинается с главного файла App.js, куда мы ссылаем Stack по навигации из файла navigation.js, в котором описана навигация между экранами.

В папке src хранятся четыре основных папки: components, data, screens, styles, как показано на рисунке 13.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13 - Основная структура приложения

Для начала рассмотрим папку screens, в которой хранятся файлы всех экранов приложения, представленную на рисунке 14.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 14 - Папка экранов приложения

* “Welcome.js” – начальный экран приложения
* “SignIn.js” – экран авторизации/регистрации пользователя
* “SignUp.js” – экран второго этапа регистрации пользователя
* “User.js” – экран учетной записи пользователя
* “ProjectsAdmin.js” – экран со списком проектов с расширенным функционалом для администратора
* “ProjectTasksAdmin.js” – экран со списком задач по проекту с расширенным фцнкционалом для администратора
* “ProjectInfo” – страница с подробной информацией о проекте (название, дедлайн, список участников)
* “ProjectTasksMember.js” – экран со списком задач по проекту
* “ProjectTaskFull.js” – экран с подробной информацией о задаче и списком подзадач

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 15 - Папка компонентов приложения

В папке components хранятся файлы, которые отвечают за разные структурные составляющие приложения, содержимое данной папки представлено на рисунке 15.

* “AddProjectTaskModal.js” – файл, отвечающий за отображение модального окна, в котором создается новая задача проекта
* “CreateProjectModal.js” – файл, отвечающий за отображение модального окна, в котором создается новый проект
* “editProjectModal.js” – файл, отвечающий за отображение модального окна, в котором редактируется выбранный проект
* “ProjectCardAdmin.js” – файл отвечающий за то, как будет отображаться карточка проекта для администратора
* “ProjectCardMember.js” – файл отвечающий за то, как будет отображаться карточка проекта для участника
* “UserCard.js” – файл отвечающий за отображение информации о пользователе на странице учетной записи

В папке “data”, которая представлена на рисунке 16, хранится один файл “firebase-config.js”, в котором подключается приложение к проекту в Firebase Console.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 16 - Папка с файлом конфигурации firebase

В папке “styles”, представленной на рисунке 17, хранятся два файла, отвечающие за стили в приложении.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 17 - Папка со стилями приложения

В файле “Colors.js” хранится цветовая гамма приложения, а в файле “styles.js” – глобальные стили компонентов приложения.

# ЗАКЛЮЧЕНИЕ

В результате проделанной курсовой работы были выполненные все поставленные задачи и создано мобильное приложение для управления проектами, основанное на методологии Kanban и использующее основной её принцип – визуализация потока Kanban-карточек на доске задач проекта.

После глубокого погружения в предметную область были определен функционал приложения, соответствующий выбранной методологии управления проектами. Разработанное приложение выполняет необходимые функции: пользователю доступна возможность регистрации и авторизации, реализовано разграничение пользователей и доступного им функционала в зависимости роли – администратора и участника, с помощью Cloud Fireastore реализован необходимый функционал по работе с данными – созданием, редактированием и удалением, так же создана функция фильтрации данных для соответствия функционала приложения методологии Kanban.

Создан простой и интуитивно понятный пользовательский интерфейс интуитивно понятен и не загружен большим количеством функций и кнопок. Приложение поддерживается мобильными устройствами на базе IOS и Android.

Данное приложение готово к работе и может обновляться добавлением новых пользовательский возможностей.

# СПИСОК ЛИТЕРАТУРЫ

1. Флэнаган Д. JavaScript. Подробное руководство. / Флэнаган Д. - СПб: Символ Плюс, 2008. - 992 с.
2. Стефанов С. JavaScript. Шаблоны. / С. Стефанов – СПб: Символ-Плюс, 2011. -272 с.
3. Официальная документация Expo. – [Электронный ресурс], URL: <https://docs.expo.dev/> (дата обращения: 01.04.2022)
4. Официальная документация React Native. – [Электронный ресурс], URL:<https://reactnative.dev/docs/getting-started> (дата обращения: 02.04.2022).
5. Информационный портал. – [Электронный ресурс], URL:<https://stackoverflow.com/> (дата обращения: 04.04.2022).
6. Сайт использованной базы данных. – [Электронный ресурс], URL:<https://firebase.google.com/> (дата обращения: 20.04.2022).
7. Образовательный портал. – [Электронный ресурс], URL:<https://koroteev.site/md/> (дата обращения: 29.04.2022).