**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |  |
| --- | --- | --- | --- |
| СОГЛАСОВАНО  Доцент департамента  Программной инженерии,  кандидат технических наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Р.З. Ахметсафина  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. | |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В. В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | RU.17701729.06.02-01 81 01-1-ЛУ | | **Работа с онлайн базой данных и функциональной частью веб-приложения для поддержки образовательного процесса “StudyGo”**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.06.02-01 81 01-1-ЛУ** | | | | | |
|  | |  | | | |
| Исполнитель  студент группы БПИ197  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Яхя Я. /  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. | | | |

**Москва 2021**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.06.02-01 81 01–1 |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | RU.17701729.06.02-01 81 01-1-ЛУ | | **Работа с онлайн базой данных и функциональной частью веб-приложения для поддержки образовательного процесса “StudyGo”**  **Пояснительная записка**  **RU.17701729.06.02-01 81 01-1**  **Листов 45** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2021**

Оглавление

[1 Введение. 3](#_Toc72262008)

[1.1 Наименование программы: 3](#_Toc72262009)

[1.2 Документы, на основании которых ведется разработка: 3](#_Toc72262010)

[2 Назначение и область применения. 4](#_Toc72262011)

[2.1 Назначение программы 4](#_Toc72262012)

[2.1.1 Функциональное назначение 4](#_Toc72262013)

[2.1.2 Эксплуатационное назначение 4](#_Toc72262014)

[2.2 Краткая характеристика области применения 4](#_Toc72262015)

[3 Технические характеристики. 5](#_Toc72262016)

[3.1 Постановка задачи на разработку программы 5](#_Toc72262017)

[3.2 Описание функционирования программы 6](#_Toc72262018)

[3.2.1 Описание функционирования различных окон программы 6](#_Toc72262019)

[3.2.2 Описание структуры онлайн базы данных Cloud Firestore 20](#_Toc72262020)

[3.3 Описание и обоснование выбора метода организации входных и выходных данных 28](#_Toc72262021)

[3.3.1 Описание метода организации входных и выходных данных 28](#_Toc72262022)

[3.3.2 Обоснование выбора метода организации входных и выходных данных 28](#_Toc72262023)

[3.4 Описание и обоснование выбора состава технических и программных средств 28](#_Toc72262024)

[3.4.1 Состав технических и программных средств 28](#_Toc72262025)

[4 Ожидаемые технико-экономические показатели 29](#_Toc72262026)

[4.1 Предполагаемая потребность 29](#_Toc72262027)

[4.2 Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 29](#_Toc72262028)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ. 30](#_Toc72262029)

[ПРИЛОЖЕНИЕ 1 ОПИСАНИЕ ФАЙЛОВ 31](#_Toc72262030)

[ПРИЛОЖЕНИЕ 2 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ 34](#_Toc72262031)

[Лист регистрации изменений. 44](#_Toc72262032)

# Введение.

## Наименование программы:

Работа с онлайн базой данных и функциональной частью веб-приложения для поддержки образовательного процесса “StudyGo”.

## Документы, на основании которых ведется разработка:

В качестве основания для разработки продукта выступает учебный план, посвященный подготовке бакалавров по направлению 09.03.04 "Программная инженерия", и тема курсового проекта, утвержденная академическим руководителем.

# Назначение и область применения.

## Назначение программы

### Функциональное назначение

Задачи:

* Авторизация пользователей.
* Сохранение данных для обеспечения выполнения логики программы и предоставление пользователям целостной информации.
* Обеспечить обмен файлами внутри приложения между пользователями.

### Эксплуатационное назначение

Веб-приложение предназначено для облегчения взаимодействия преподавателей и студентов. Взаимодействие в следующем: преподаватели публикуют объявления и задания, а студенты отправляет выполненные работы в ответ на задания для обязательного выполнения. Преподаватели также выдают студентам результат проверки принятых от студентов работ.

## Краткая характеристика области применения

Текущая программа будет полезна в учебных заведениях для преподавателей и студентов в выдаче заданий и отправке решений, которые потом оцениваются преподавателями, соответственно.

# Технические характеристики.

## Постановка задачи на разработку программы

Для всех пользователей:

* + Предоставить возможность авторизации пользователя и сохранение его в базе данных.
  + Предоставить возможность создания группы и сохранение ее в базе данных.
  + Предоставить возможность создания каналов в каждой учебной группе, связанных с конкретными дисциплинами, и сохранение их в базе данных.
  + Приглашение и добавление пользователя в конкретную учебную группу по ее уникальной ссылке и сохранение его участия в группе в базе данных.
  + Обеспечить корректный показ ленты сообщений в конкретном канале на основании данных из базы данных.
  + Корректный показ ленты всех заданий для необязательного выполнения со всех каналов на основании данных из базы данных.
  + Корректный показ ленты всех заданий для обязательного выполнения со всех каналов на основании данных из базы данных.
  + Предоставить возможность загрузки выполненных работ в ответ на сообщения с обязательным заданием и сохранение файлов-решений в хранилище данных, а их URL-ссылок в базе данных.
  + Корректный показ обратной связи в виде оценок и комментариев от преподавателей на основании данных из базы данных.
  + Предоставить возможность сохранения новых названия и картинки группы после редактирования.
  + Предоставить возможность покинуть группу и удалить информацию о группе из базы данных.

Преподаватель:

* + Предоставить возможность создания публикаций и сохранения составных данных публикации в базе данных.
  + Предоставить возможность предоставления рецензии по выполненным работам от преподавателей и сохранения этих рецензий в базе данных.
  + Предоставить показ списка отправленных студентами работ в ответ на задания для обязательного выполнения от преподавателя.

## Описание функционирования программы

### Описание функционирования различных окон программы

#### Окно авторизации

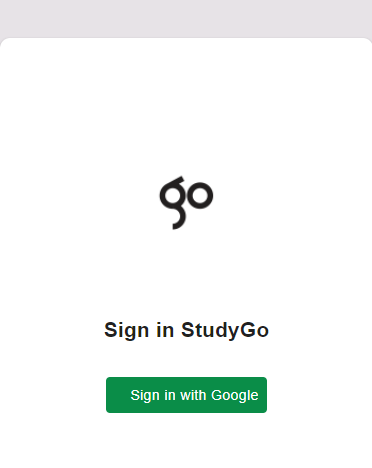


Рисунок 1 - Окно авторизации

Для авторизации пользователей используется служба Firebase Authentication с помощью google аккаунта. Управление учетными записями происходит в личном кабинете Firebase Authentication (см. рисунок 2). Для работы с этой службой используется модуль react-firebase-hooks/auth при помощи которого можно получить доступ к данным Google-account пользователя.

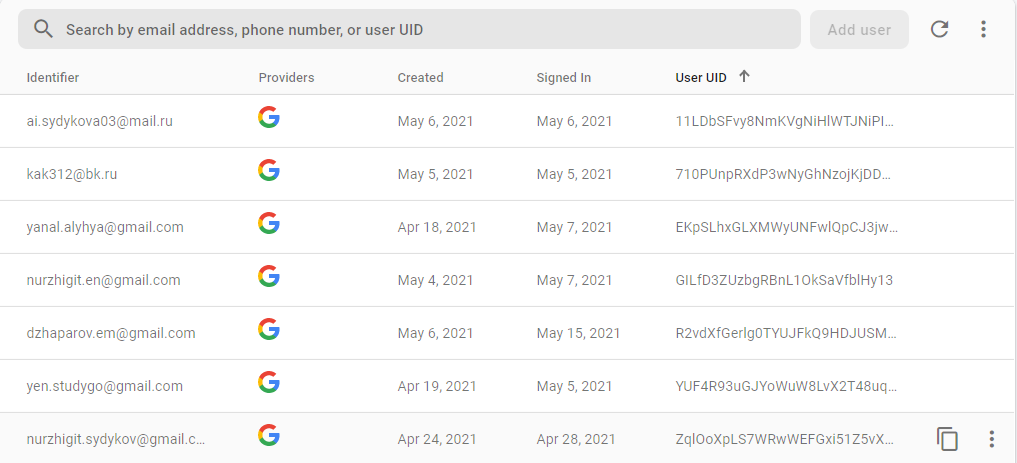


Рисунок 2. Управление учетными записями авторизованных.

#### Панель приложения



Рисунок 3 - Панель приложения

Панель приложения (см. рисунок 3) получаем из файла App.js, извлекаем группу при помощи хука useDocument из модуля react-firebase-hooks/firestore, а пользователя при помощи хука useAuthState из модуля react-firebase-hooks/auth, где группа представлена в виде документа (рисунок 4), важно отметить, что при извлечении данных происходит загрузка только полей, без коллекций. Пользователь представляет объект, который инкапсулирует информацию из учетной записи Google.

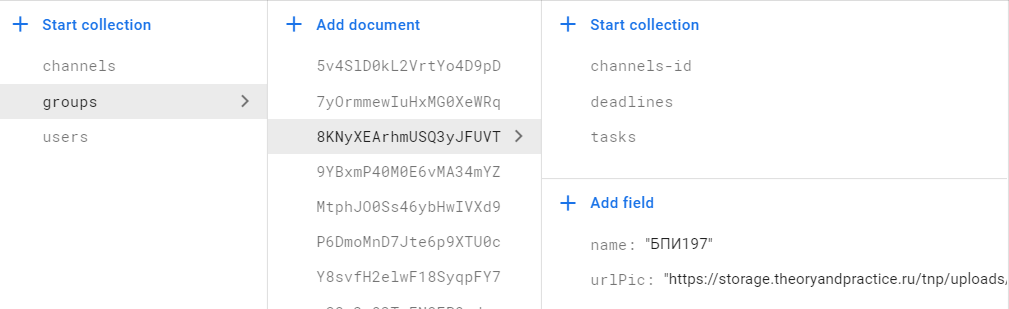


Рисунок 4. Представление групп в firestore

#### Панель навигации

Компоненту панели навигации получаем из файла Sidebar.js. Извлекаем хуком useDocument группу (рисунок 4) и пользователя (рисунок 5). По нажатию на кнопку Tasks переходим в **окно необязательных задач**, а по нажатию на кнопку Deadlines в **окно обязательных заданий**. По нажатию на кнопку Add Channel, открывается **окно создания канала**. Извлекаем коллекцию идентификаторов каналов группы (рис) и по нажатию по названию канала переходим в **окно канала**, сохранив в Redux идентификатор канала.

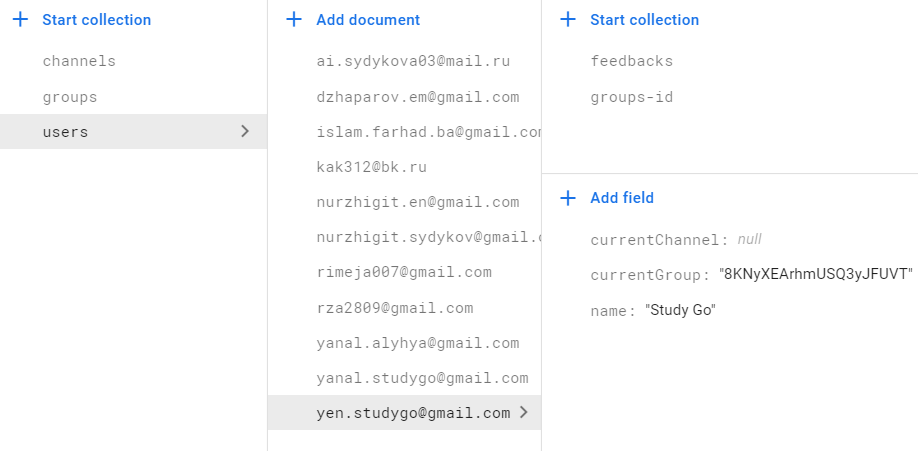
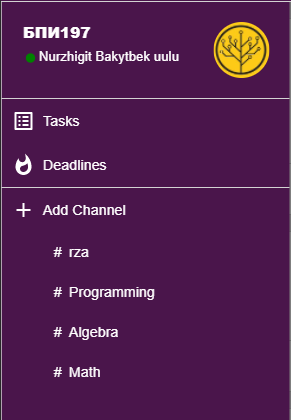
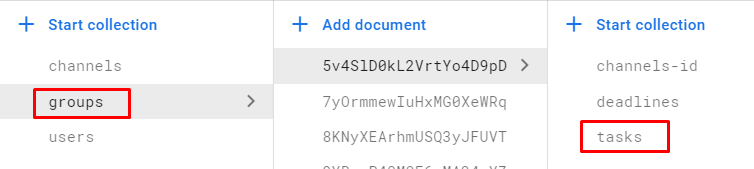


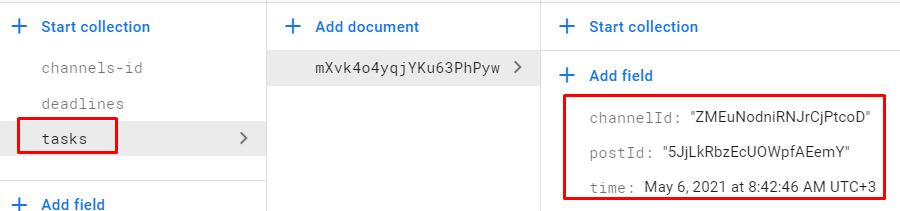
Рисунок 4. Представление пользователей в firestore

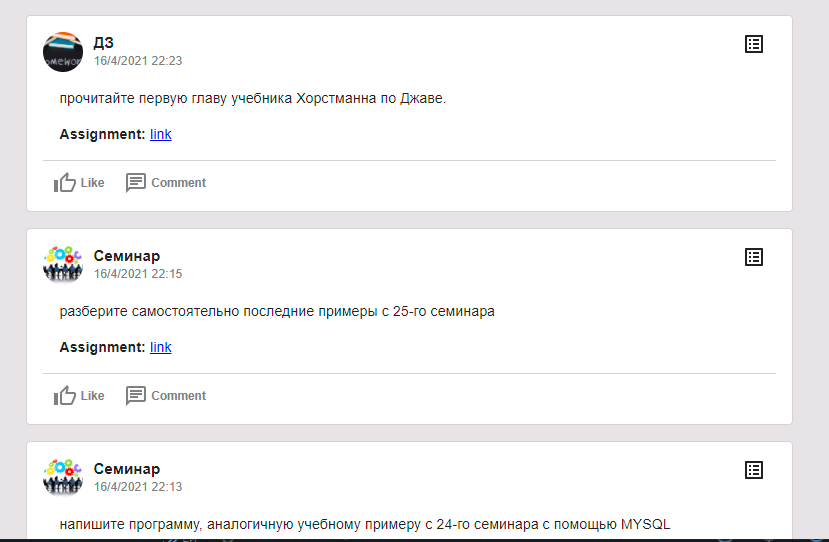
#### Окно просмотра всех необязательных (без крайнего срока) заданий

  
Рисунок 6 - Панель навигации

Компоненту необязательных заданий получаем из файла Tasks.js. Каждое задание представлено компонентой из файла TaskItem.js. В Tasks.js мы извлекаем данные при помощи модуля react-firebase-hooks/firestore мы извлекаем коллекцию tasks группы (рисунок 7), которые содержат необходимые идентификаторы для представления задания, потом мы циклом отображаем эту коллекцию в виде компоненты из TaskItem.js, где извлекаем. Коллекцию tasks сортируем по неубыванию времени добавления. При создании необязательного задания внутри любого канала в коллекцию tasks группы добавляем новый элемент.



  
Рисунок 7- Представление необязательных заданий в firestore

  
Рисунок 8- Необязательные заданий

#### Окно просмотра всех обязательных (c крайним сроком) заданий

Компоненту необязательных заданий получаем из файла Deadlines.js. Каждое задание представлено компонентой из файла TaskItem.js. В Tasks.js мы извлекаем данные при помощи модуля react-firebase-hooks/firestore мы извлекаем коллекцию deadlines группы (рисунок 8), которые содержат необходимые идентификаторы для представления задания, потом мы циклом отображаем эту коллекцию в виде компоненты из TaskItem.js, где извлекаем. Коллекцию deadlines сортируем по неубыванию времени добавления. При создании обязательного задания внутри любого канала в коллекцию deadlines группы добавляем новый элемент.

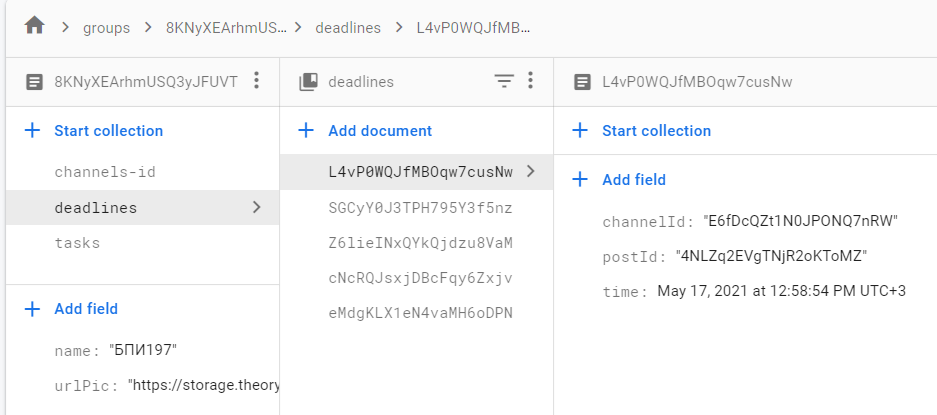
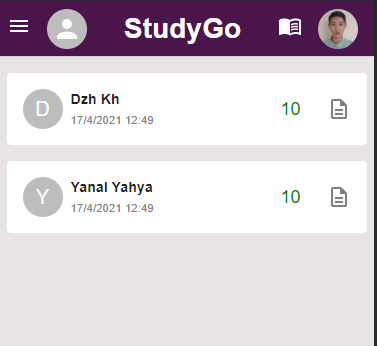
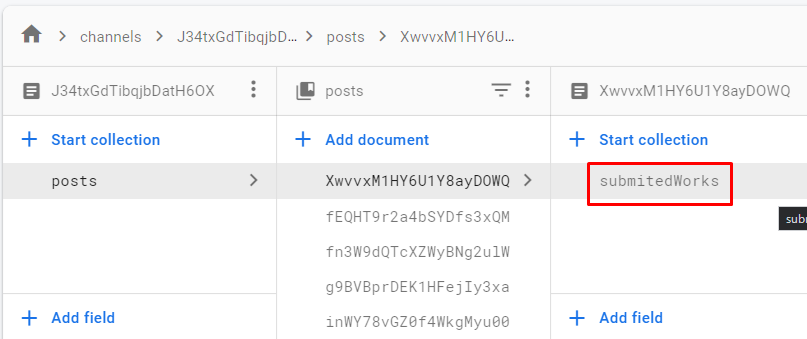


Рисунок 8- Представление обязательных заданий в firestore

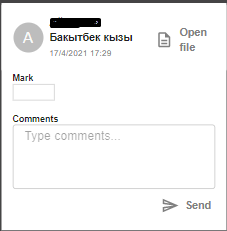
#### Окно со списком отправленных выполненных работ от обучающихся

  
Рисунок 9 – Присланные работы от обучающихся

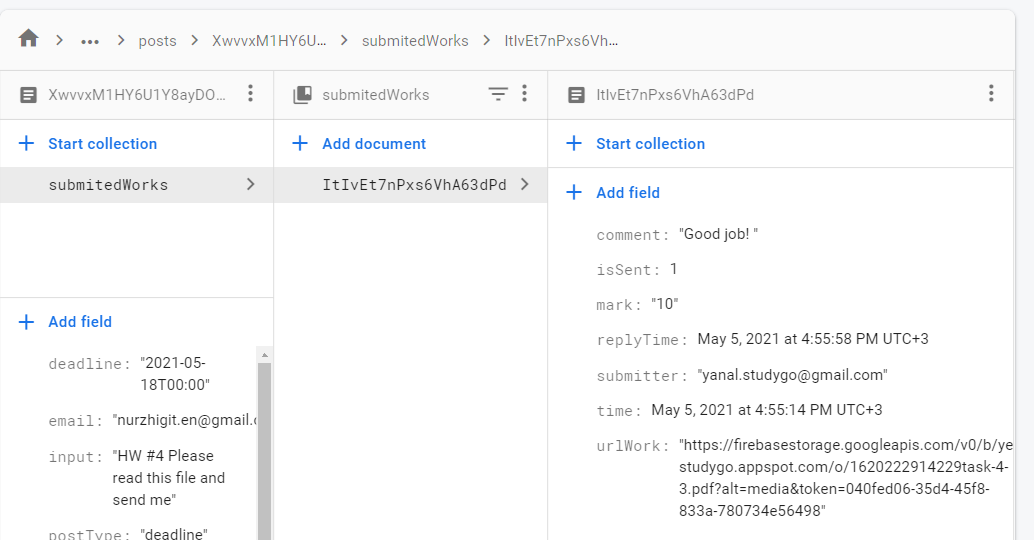
Окно со списком отправленных выполненных работ от обучающихся получаем из файла Submits.js, где при помощи хука useCollection получаем submitedWorks (рисунок 9–10). Элменты этой коллекции отображаем в виде компоненты из файла WorkItem.js.

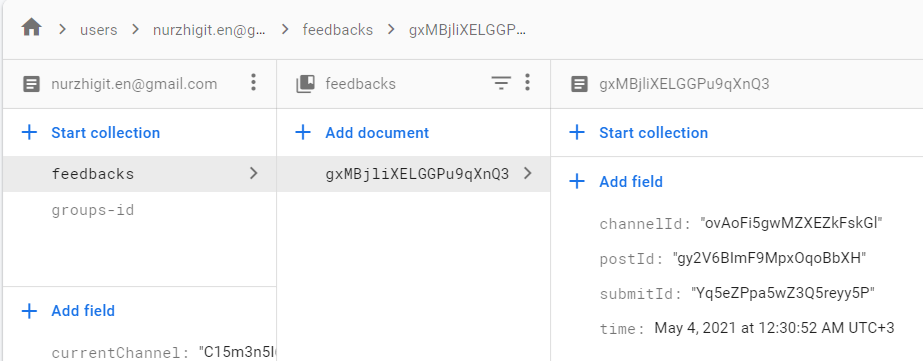
  
Рисунок 10 - Представление присланных работ от обучающихся в firestore

#### Вспомогательное окно создания обратной связи

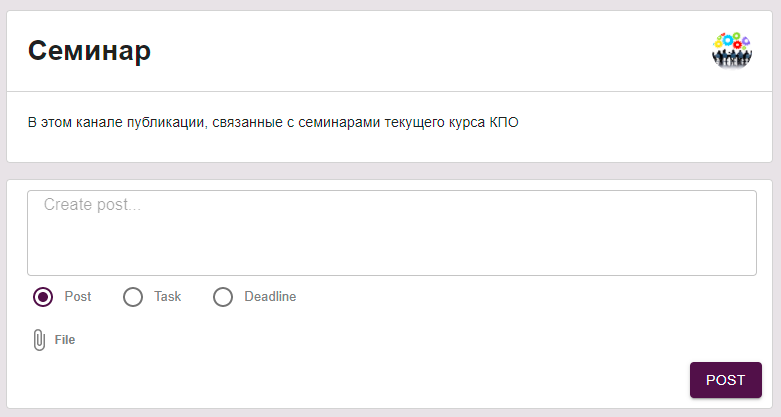
  
Рисунок 11 - Вспомогательное окно создания обратной связи

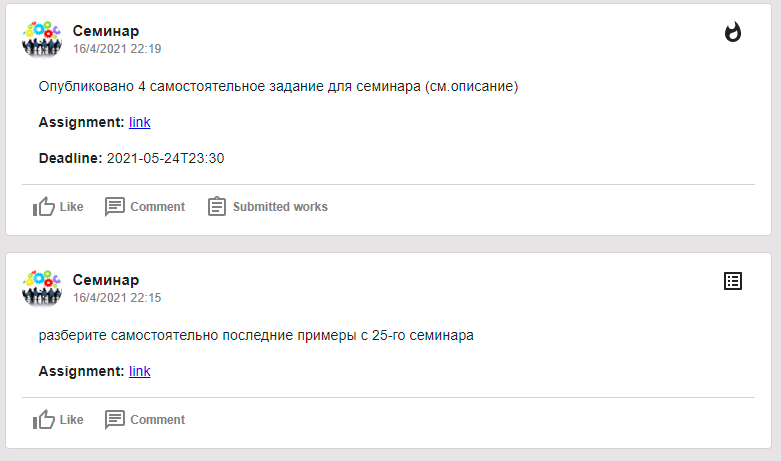
Вспомогательное окно создания обратной связи (рисунок 11) получаем из файла WorkItemModal, получаем данные от пользователя и добавляем в коллекцию submitedWorks и добавляем к отправителю в коллекцию feedbacks необходимые данные (см. рисунок 12-13).

  
Рисунок 12 – Представление присланных работ от обучающихся в firestore

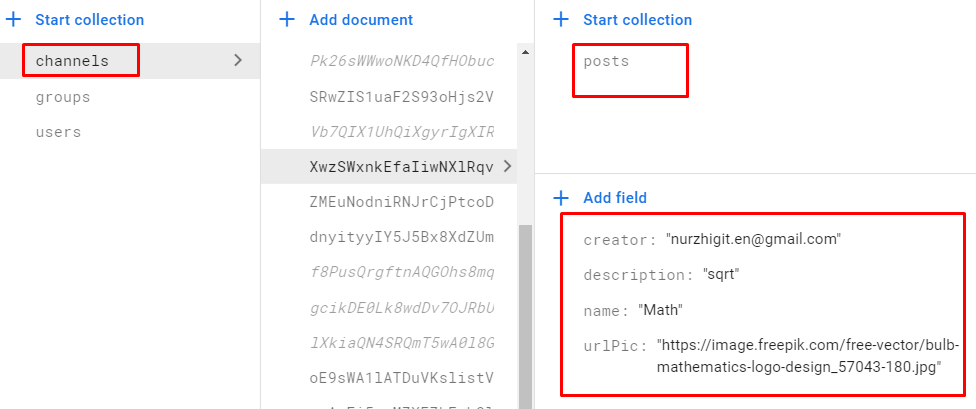
  
Рисунок 13 – Представление обратных связей от рецензентов у студента в firestore

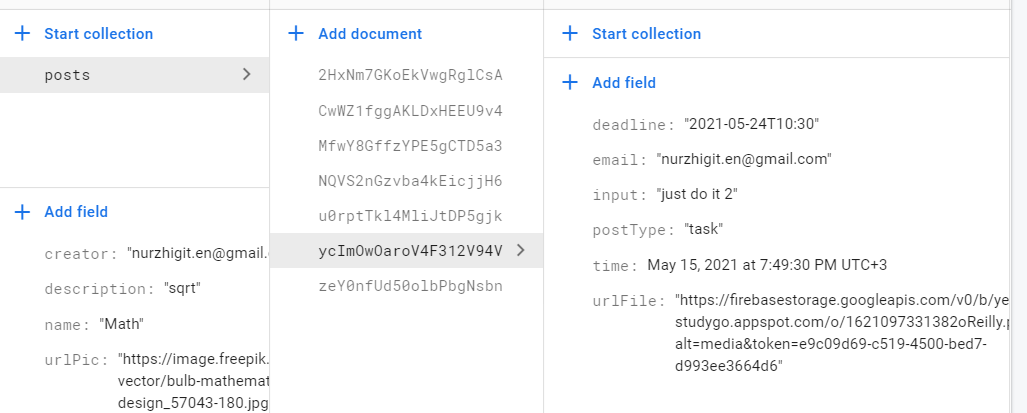
#### Окно просмотра ленты конкретного канала



  
Рисунок 14 – лента конкретного канала

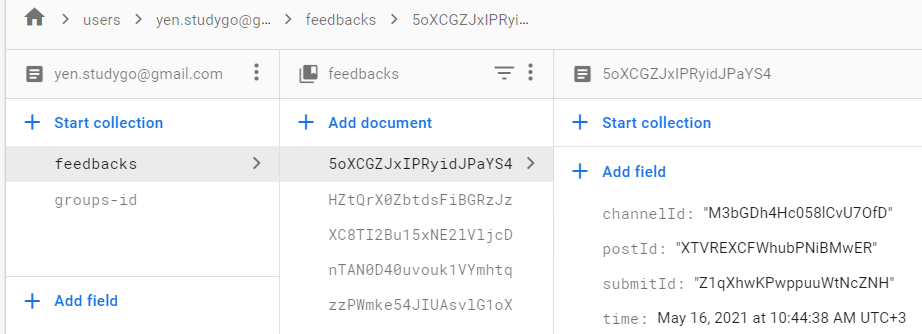
Окно просмотра ленты получаем из файла ChannelPage.js, получаем пользователя хуком useAuthState, получаем идентификатор канала и группы из Redux при помощи хуков useSellector из модуля react-redux. Получаем по идентификатору канала элемент из коллекции channels. У этого элемента при помощи хука useCollection загружаем коллекцию posts. Циклом по элементам (рисунок 14) коллекции posts отображаем элементы в виде компонент из файла PostItem.js.



  
Рисунок 15 – Представление всех типов публикацией в firestore

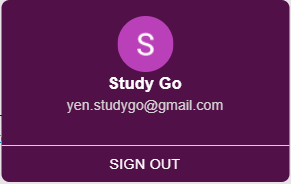
#### Окно пользователя для просмотра списка оценок и комментариев от преподавателей:

Окно оценок и комментариев от преподавателей получаем из файла Feedbacks.js при помощи Redux получаем элемент из коллекции пользователей (рисунок 16) и из него извлекаем коллекцию feedbacks. Элементы этой коллекции отображаем в виде компоненту из файла FeedbackItem.js, где извлекаем данные при помощи useDocument.

  
Рисунок 16 – Представление обратных связей от рецензентов у студента в firestore

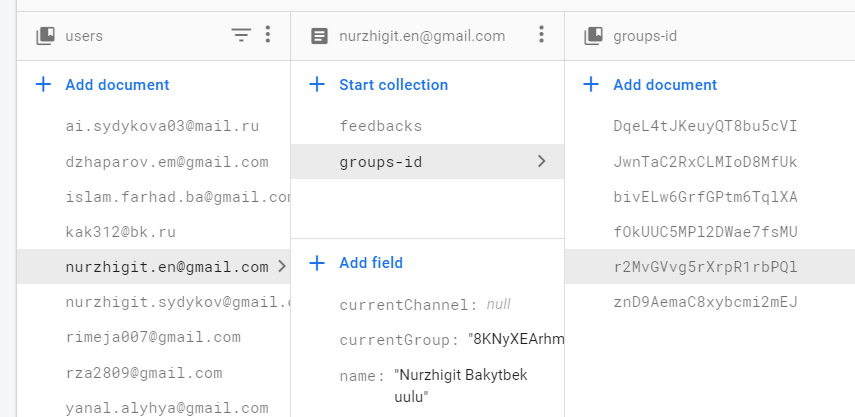
#### Выпадающее окно выхода из аккаунта

При нажатии на кнопку SIGN OUT, выходим из аккаунта при помощи объекта auth из файла firebase.js в файле UserSettingSidebar.js (см рисунок 17).

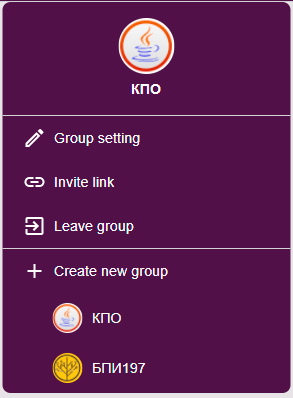
  
Рисунок 17 – Выпадающее окно выхода из аккаунта

#### Выпадающее окно управления группой

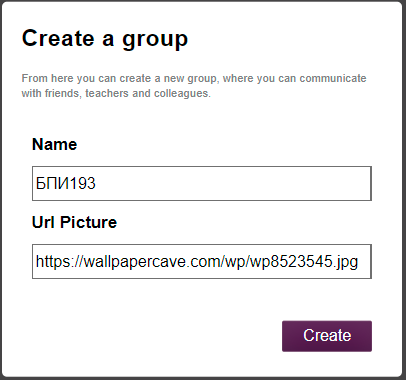
При помощи Redux и хуков получаем текущую группу из коллекции groups. При нажатии на кнопку Leave group удаляем элемент из коллекции groups-id пользователя (рисунок 18)

  
Рисунок 18 – Представление коллекции groups-id у пользователя в firestore

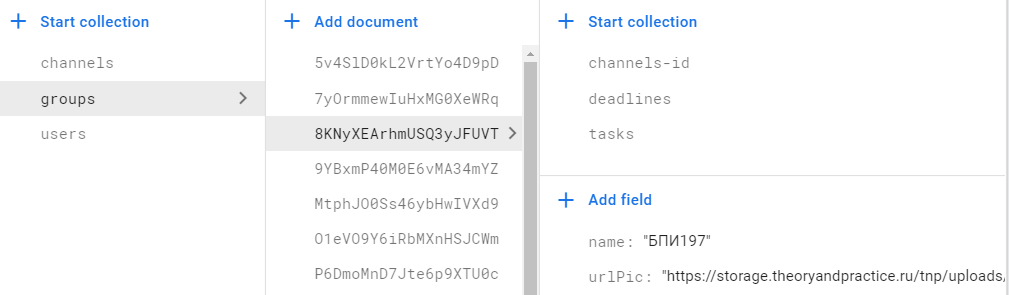
При нажатии на иконку или название определенной группы мы меняем состояние в Redux и в поле currentGroup у пользователя.

  
Рисунок 19 – Панель управление группы

#### Вспомогательное окно создания группы содержит:

  
Рисунок 20 – Вспомогательное окно создания группы

Вспомогательное окно создания группы получаем из файла CreateGroupModal.js. Получаем данные от пользователя и создаем группу в коллекции groups (рисунок 21).

  
Рисунок 21 – Представление коллекции групп в firestore

#### Вспомогательное окно для редактирования настройки группы.

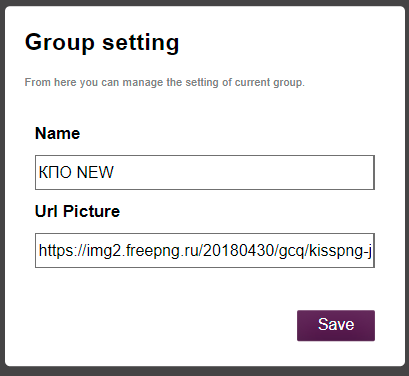
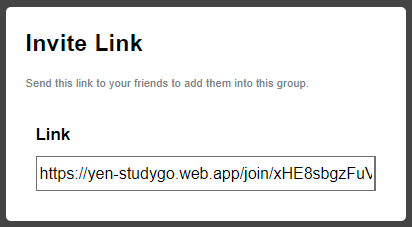


Рисунок 22 - Вспомогательное окно для редактирования настройки группы

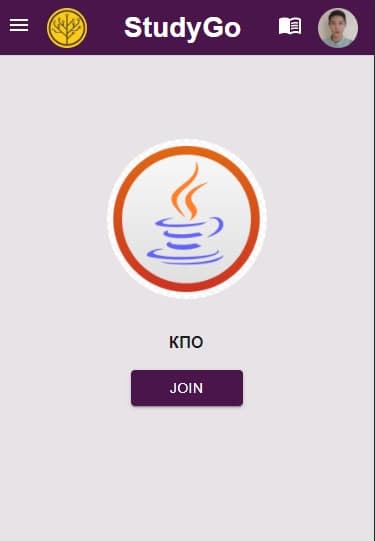
Вспомогательное окно для редактирования настройки группы (рисунок 22) получаем из файла EditGroupSettingModal, где мы получаем новые данные и обновляем элемент в коллекции groups.

**Вспомогательное окно для получения ссылки для присоединения к группе**

Берется идентификатор текущей группы и перед ним добавляется домен приложения и “/join/”.

  
Рисунок 23 – Вспомогательное окно для получения ссылки для присоединения к группе

#### Окно присоединения к группе

  
Рисунок 24 – Окно присоединения к группе

Окно присоединения к группе получаем из файла JoinPage.js. Получаем из адресной строки идентификатор группы и добавляем к коллекции groups-id (рисунок 18).

### Описание структуры онлайн базы данных Cloud Firestore

В текущей работе использовалась нереляционная база данных Cloud Firestore. Настоящая база данных выделяется документно-ориентированным подходом в хранении информации, то есть от реляционных баз данных она отличается тем, что в ней нет таблиц или строк. Данные хранятся в документах, которые, в свою очередь, распределены в коллекции.

Каждый документ содержит пары из ключа и значения. В каждом файле информация хранится в JSON-формате. Каждый документ содержится в коллекциях. Документ может содержать внутри себя вложенные коллекции и вложенные объекты, которые могут иметь в себе поля, строки или объекты, например списки.

**КОЛЛЕКЦИЯ ПОЛЬЗОВАТЕЙ**

Коллекция пользователей (рисунок 25) — это набор пользователей, каждый из которых представлен в виде пары уникальный идентификатор пользователя и данные, ассоциированные с пользователем.

**ОБЪЕКТ-ПОЛЬЗОВАТЕЛЬ**

Главный объект в базе данных — это пользователь, у которого имеются свои поля и коллекции.

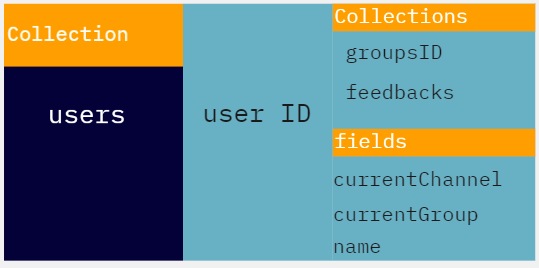


Рисунок 25 - Коллекция пользователей

Поля пользователя:

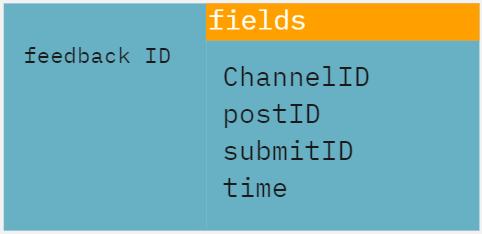
* Текущая группа, с которой работает пользователь в данный момент. Группа представлена в виде идентификатора — это сделано для того, чтобы запомнить последнюю группу, который посетил пользователь, то есть реализовано сохранение последнего сеанса.
* Текущий канал, с содержанием которого работает пользователь в данный момент. Канал представлен в виде идентификатора — это сделано для того, чтобы запомнить последний канал, который посетил пользователь.
* Имя пользователя.

Коллекции пользователя:

* Группы (рисунок 26), в которых состоит пользователь. Представлены в виде уникальных идентификаторов, через которые можно получить доступ к соответствующим объектам - группам.

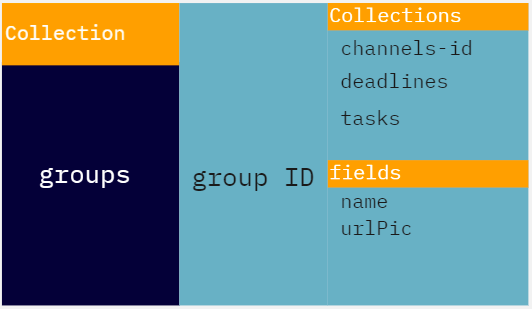
  
Рисунок 26 – Коллекция идентификаторов групп

* Обратные связи (оценки, комментарии) от преподавателей представлены в виде идентификаторов (рисунок 27). Внутри каждого объекта - обратной связи содержатся:
  + - Идентификатор канала, к которому относится эта обратная связь. С помощью идентификатора канала происходит обращение к коллекции каналов для получения самого объекта канала.
    - Идентификатор обязательного задания, на которое высылали выполненную работу, к которой была создана исходная обратная связь.
    - Идентификатор высланной выполненной работы. С помощью идентификаторов обязательного задания канала и отправленной работы можно получить саму отправленную выполненную работу.
    - Время создания обратной связи. С помощью этого параметра происходит сортировка всех объектов - обратных связей.

  
Рисунок 27 – Коллекция идентификаторов, которые помогут достать submit элемент

**КОЛЛЕКЦИЯ ГРУПП**

Коллекция групп (рисунок 28) - то набор объектов - групп, каждый из которых представлен в виде пары уникальный идентификатор группы и данные, ассоциированные с группами.

  
Рисунок 28 - Коллекция групп

**ОБЪЕКТ - ГРУППА**

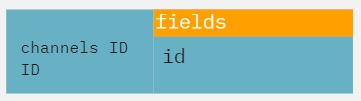
Объект-группа имеет следующие поля и коллекции:

Поля группы:

* Название группы.
* URL-ссылка на картинку группы (аватар).

Коллекции группы:

* Каналы (рисунок 29), которые находятся в этой группе. Представлены в виде уникальных идентификаторов, через которые можно получить доступ к соответствующим объектам - каналам.

  
Рисунок 29 – Коллекция идентификаторов каналов

* Обязательные задания (Рисунок 30) от преподавателей представлены в виде идентификаторов. Внутри каждого объекта содержатся:
  + Идентификатор канала, к которому относится это обязательное задание. С помощью идентификатора канала происходит обращение к коллекции каналов для получения самого объекта канала.
  + Идентификатор обязательного задания, с помощью которого можно получить сам объект - сообщение с публикацией.
  + Время создания обязательного задания. С помощью этого параметра происходит сортировка всех объектов - обязательных заданий.

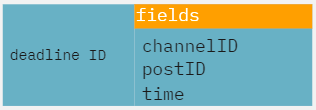


Рисунок 30 – Коллекция идентификаторов, которые помогут достать обязательное задание

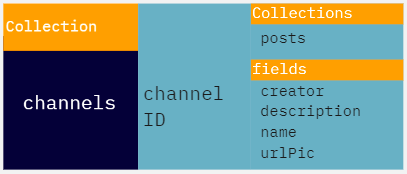
* Необязательные задания (рисунок 31) от преподавателей представлены в виде идентификаторов. Внутри каждого объекта содержатся:
  + - Идентификатор канала, к которому относится это необязательное задание. С помощью идентификатора канала происходит обращение к коллекции каналов для получения самого объекта канала.
    - Идентификатор необязательного задания, с помощью которого можно получить сам объект - задание.
    - Время создания необязательное задания. С помощью этого параметра происходит сортировка всех объектов - необязательных заданий.



Рисунок 31 – Коллекция идентификаторов, которые помогут достать необязательное задание

**КОЛЛЕКЦИЯ КАНАЛОВ**

Коллекция каналов (рисунок 32) — это набор объектов - каналов, каждый из которых представлен в виде пары уникальный идентификатор канала и данные, ассоциированные с каналами.

  
Рисунок 32 – Коллекция каналов

**ОБЪЕКТ-КАНАЛ**

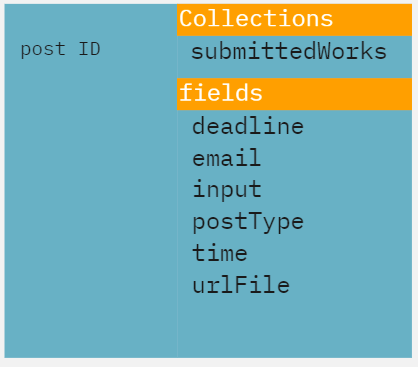
Объект-канал имеет следующие поля и коллекции:

Поля канала:

* Google-почта создателя канала, через который он авторизовался.
* Описание канала.
* Название канала.
* URL-ссылка на картинку канала (аватар)

Коллекции канала (рисунок 33):

* Объекты - публикации, которые состоят из своих идентификаторов и данных:

  
Рисунок 33 – коллекция публикацией

Поля канала:

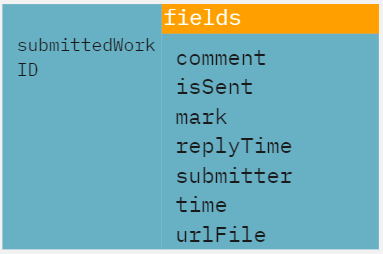
* Крайний срок сдачи работы.
* Почта пользователя, который создал публикацию.
* Текстовое содержание публикации.
* Тип публикации (сообщения): объявления, обязательные либо необязательное задание.
* Время публикации, по которому осуществляется сортировка.
* URL - ссылка на приложенный файл, которая генерируется через специальный скрипт в проекте после сохранения файла в Firebase Storage.

Коллекции канала:

* Отправленные обучающимися выполненные работы. Они показываются только у публикаций с обязательным заданием.

**КОЛЛЕКЦИЯ-ОТПРАВЛЕННОЕ РЕШЕНИЕ**

Коллекция отправленных решений (Рисунок 34) — это набор объектов - отправленных решений, каждый из которых представлен в виде пары уникальный идентификатор решения и данные, ассоциированные с отправленными решениями. Имеются только у публикаций с обязательными заданиями.

  
Рисунок 34 – Коллекция отправленных решений

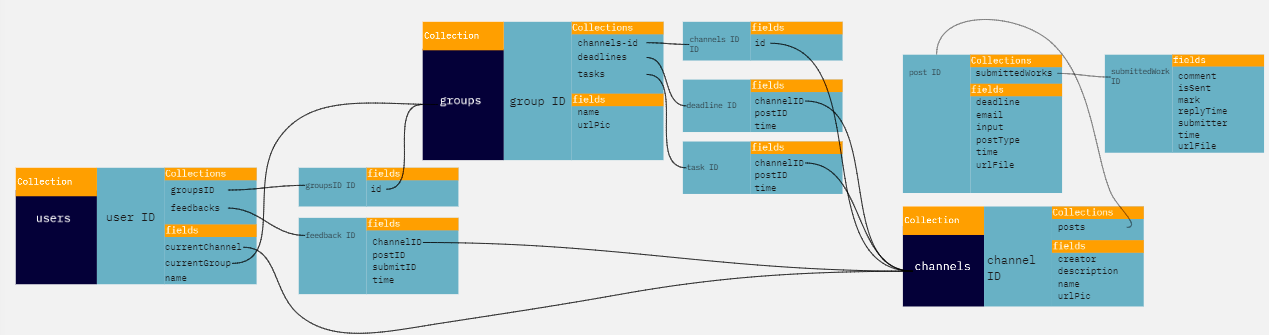
**ОБЪЕКТ-ОТПРАВЛЕННОЕ РЕШЕНИЕ**

Объект состоит из следующих полей и коллекций:

Поля канала:

* Комментарий по выполненной работе от проверяющего.
* Индикатор, который сообщает о том, отправлена ли обратная связь или нет.
* Оценка по выполненной обучающимся работе.
* Время отправки обратной связи.
* Email обучающегося, который отправил работу.
* Время отправки работы обучающимся на проверку.
* URL - ссылка на отправленный файл для проверки, которая генерируется через специальный скрипт в проекте после сохранения файла в Firebase Storage.

На рисунке 35 представлены все вышеперечисленные объекты и связи между ними:

  
Рисунок 35 – представление все вышеперечисленные объекты и связи между ними

## Описание и обоснование выбора метода организации входных и выходных данных

### Описание метода организации входных и выходных данных

Входными данными являются действия пользователей (нажатия на кнопки), тексты, получаемые из полей ввода и информации, получаемы от СУБД Firestore и хранилища Firebase storage.

Выходными данными являются информация, отправляемые в СУБД (Firebase Realtime Database) и хранилища Firebase storage.

### Обоснование выбора метода организации входных и выходных данных

Облачная СУБД Firebase Firestore выбран из следующих соображений:

* Дает возможность синхронизировать и хранить информацию пользователей в режиме реального времени.
* Нет потребности создания сервера.
* Пользовательская безопасность.

## Описание и обоснование выбора состава технических и программных средств

### Состав технических и программных средств

Для корректной работы приложения нужен мобильный телефон или ПК с доступом в интернет со следующим набором свойств:

* Количество ядер - 4
* Частота процессора – 1.2 ГГц - для телефона, 1.4 ГГц - для ПК
* Объем ОЗУ - 2 ГБ
* Объем ПЗУ - 16 ГБ
* ОС с установленным интернет-браузером Google Chrome с версией 62.0.3202.84 и выше.

# Ожидаемые технико-экономические показатели

## Предполагаемая потребность

Это приложение нацелено на студентов и на преподавателей школ и университетов. С данной программой будет обеспечен удобный процесс выдачи заданий преподавателями, сдачи и приема работ студентов, их проверка.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Этот проект выполнен для получения опыта и повышения навыков разработчиков, экономических целей нет.

Преимуществами программы являются:

* Является бесплатным.
* Возможность работы с другими пользователями.
* Не требует денежных затрат для выполнения проекта.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Модель данных Cloud Firestore [Электронный ресурс]// URL: <https://firebase.google.com/docs/firestore/data-model> (Дата обращения 26.3.2021, режим доступа: свободный).
2. SDK и клиентские библиотеки [Электронный ресурс]// URL:<https://firebase.google.com/docs/firestore/client/libraries> (Дата обращения 26.3.2021, режим доступа: свободный).
3. Добавить данные в Cloud Firestore [Электронный ресурс]// URL:<https://firebase.google.com/docs/firestore/manage-data/add-data> (Дата обращения 28.4.2021, режим доступа: свободный).
4. Удалить данные из Cloud Firestore [Электронный ресурс]// URL: <https://firebase.google.com/docs/firestore/manage-data/delete-data> (Дата обращения 28.4.2021, режим доступа: свободный).
5. Управление Cloud Firestore с помощью консоли Firebase [Электронный ресурс]// URL: <https://firebase.google.com/docs/firestore/using-console> (Дата обращения 3.4.2021, режим доступа: свободный).
6. 10 часов видео на английским языке для изучение ReactJS [Электронный ресурс]// URL: <https://youtu.be/4UZrsTqkcW4> (Дата обращения 29.2.2021, режим доступа: свободный).
7. Redux For Beginners | React Redux Tutorial [Электронный ресурс] // URL: <https://youtu.be/CVpUuw9XSjY> (Дата обращения 3.4.2021, режим доступа: свободный).

# ПРИЛОЖЕНИЕ 1 ОПИСАНИЕ ФАЙЛОВ

Таблица П1 - Описание и функциональное назначение файлов

|  |  |
| --- | --- |
| Файл | Назначение |
| CheckPage.js | Реализация страницы с обратными связями от преподавателей |
| Submits.js | Представление страницы со списком отправленных обучающимися решений. Страница формируется через извлечение данных, хранящихся в Firestore. |
| WorkItem.js | Графическое представление объекта отправленной работы |
| WorkItem.css | Стили к WorkItem.js элементам |
| WorkItemModal.js | Реализует окно, в котором учитель ставит оценку, комментарий и отправляет их учащемуся. |
| FeedbackItem.js | Графическое представление объекта обратной связи от проверяющего |
| Feedbacks.js | Представление страницы со списком отправленных обучающемуся обратных связей от преподавателей. Страница формируется через извлечение данных, хранящихся в Firestore. |
| CreateGroupModal.js | Реализует вспомогательное окно, через которое можно ввести данные о новой группе и создать ее. |
| EditGroupSettingModal.js | Реализует вспомогательное окно, где можно редактировать название и URL-ссылку на аватар текущей группы. |
| GroupSettingMenu.js | Реализует выпадающее окно управления группой. |
| GroupSettingMenu.сss | Стили к GroupSettingMenu.js элементам. |
| InviteLink.js | Реализует вспомогательное окно, где можно скопировать ссылку-приглашение, через которую можно вступить в группу. |
| ChannelBuildPost.js | В окне канала реализует блок, через который можно сделать публикацию трех типов: объявление и 2 типа заданий. |
| ChannelBuildPost.css | Стили к ChannelBuildPost.js элементам. |
| ChannelHeader.js | В окне канала реализует блок, где есть название канала, аватар канала и описание канала. |
| ChannelHeader.css | Стили к ChannelHeader.js элементам. |
| ChannelPage.js | Представление страницы канала. |
| Deadlines.js | Представление страницы со списком всех обязательных заданий от преподавателей. Страница формируется через извлечение данных, необходимых для формирования блоков обязательных заданий, хранящихся в Firestore. |
| FileAttachModal.js | Выпадающее окно, в котором преподаватель может прикрепить файл для публикации. |
| HomePage.js | Окно, в котором показаны обзор приложения, поясняющее видео и информация о разработчиках. |
| HomePage.css | Стили к HomePage.js элементам. |
| InputOption.js | Объект Icon-Title в виде кнопки. Альтернативно, пункты выпадающего меню управления группой. |
| InputOption.css | Стили к InputOption.js элементам. |
| JoinPage.js | Окно, через которое пользователь может вступить в группу. |
| JoinPage.css | Стили к JoinPage.js элементам. |
| PostItem.js | Реализует три типа блоков публикаций. |
| PostItem.css | Стили к PostItem.js элементам. |
| SubmitPopUp.js | Вспомогательное окно, где студент может загрузить свой файл с решением в ответ на задание. |
| TaskItem.js | Реализует блок необязательного задания. |
| Tasks.js | Извлечение данных, необходимых для необязательного задания, из firestore и представление этих данных в виде блока TaskItem. |
| AddChannelPopUp.js | Выпадающее окно, через которое можно добавить новый канал, введя его название, описание и ссылку на картинку канала. |
| AddChannelPopUp.css | Стили к AddChannelPopUp.js элементам. |
| Bottom.js | Представление нижней части панели навигации. |
| Bottom.css | Стили к Bottom.js элементам. |
| Middle.js | Представление средней части панели навигации. |
| Middle.css | Стили к Middle.js элементам. |
| SidebarOption.js | Объект Icon-Title в виде кнопки для панели навигации, ассоциирующийся с обязательными или необязательными заданиями. |
| SidebarSubject.js | Объект-канал в списке каналов внутри панели навигации. |
| StudyGoLogo.js | Логотип приложения. |
| Top.js | Верхняя часть панели навигации. |
| Top.css | Стили к Top.js элементам. |
| UserSettingSidebar.js | Окно, через которое можно выйти из аккаунта. |
| Login.js | Окно авторизации. |
| MyAppBar.js | Панель приложения. |
| Sidebar.js | Панель навигации. |
| Sidebar.css | Стили к Sidebar.js элементам. |
| firebase.js | Инициализация и Извлечение объекты от модуля firebase |

# ПРИЛОЖЕНИЕ 2 ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ

Таблица П2 - Описание функций файла CheckPage.js

|  |  |
| --- | --- |
| Методы | |
|  | Назначение |
|  | Возвращает страницу проверки, где внутри тега вызывается Submits.js (который возвращает список работ, которые были отправлены от обучающихся) |

Таблица П3 - Описание методов и параметров файла Submits.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Submits | Возвращает список работ, которые были отправлены от обучающегося, в виде отдельных блоков. |

Таблица П4 - Описание методов и параметров файла WorkItem.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| WorkItem | Реализует отдельный блок присланной работы. |
| Параметры | |
| Имя | Назначение |
| id | ID присланной работы от обучающегося. |
| submitDoc | Ссылка на данные, которые ассоциируются вместе с присланной работой. |

Таблица П5 - Описание методов и параметров файла WorkItemModal.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| WorkItemModal | Реализует окно, в котором рецензент может оценивать и комментировать принятую от студента работу и отправлять результаты обратно обучающемуся. |
| Параметры | |
| Имя | Назначение |
| id | ID присланной работы обучающегося. |
| onClose | Функция для закрытия этого окна. |
| submitter | Ссылка на данные человека, отправившего работу. |
| date | Время, когда работа была прислана от обучающегося. |
| submit | Ссылка на данные, поступившие вместе с присланной работы. |

Таблица П6 - Описание методов и параметров файла FeedbackItem.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| FeedbackItem | Реализует отдельный блок обратной связи в окне обратных связей у обучающегося. |
| Параметры | |
| Имя | Назначение |
| channelId | Идентификатор канала, в котором было опубликовано обязательное задание. |
| postId | Идентификатор публикации, опубликованной внутри канала. |
| submitId | Идентификатор присланной работы от обучающегося. |
| time | Окончательный срок сдачи обязательного задания. |

Таблица П7 - Описание методов и параметров файла Feedbacks.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Feedbacks | Возвращает список всех проверенных работ и обратной связи по ним от рецензента для студента после их извлечения из firestore. |

Таблица П8 Описание методов и параметров файла CreateGroupModal.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| CreateGroupModal | Реализует выпадающее окно, где можно заполнять необходимые данные для создания новой группы и, собственно, создать её. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция, для закрытия этого окна. |

Таблица П9 - Описание методов и параметров файла EditGroupSettingModal.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| EditGroupSettingModal | Реализует выпадающее окно, где можно редактировать название и URL-фото выбранной группы. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция, для закрытия этого окна. |

Таблица П10 - Описание методов и параметров файла GroupSettingMenu.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| GroupSettingMenu | Реализует выпадающее окно управления группой. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция, для закрытия этого окна. |
| groupSetting | Сигнал для открытия выпадающего окна управления группой, если он не null. |

Таблица П11 - Описание методов и параметров файла InviteLink.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| InviteLink | Реализует вспомогательное окно, где можно скопировать ссылку, через которую потом можно вступить в группу. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция, для закрытия этого окна. |

Таблица П12 - Описание методов и параметров файла ChannelBuildPost.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| ChannelBuildPost | Реализует блок, через который можно сделать публикацию. |

Таблица П13 - Описание методов и параметров файла ChannelHeader.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| ChannelHeader | Реализует блок, в котором присутствует название канала, аватар канала и описание канала. |
| Параметры | |
| Имя | Назначение |
| name | Название канала. |
| description | Описание канала. |
| urlPic | URL-ссылка на картину(аватар). |

Таблица П14- Описание методов и параметров файла ChannelPage.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| ChannelPage | Реализует страницу канала. |

Таблица П15 - Описание методов и параметров файла Deadlines.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Deadlines | Извлечение данных, связанных с обязательной работой, из firestore. |

Таблица П16 - Описание методов и параметров файла FileAttachModal.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| FileAttachModal | Реализует вспомогательное окно, в котором можно прикрепить файл для публикации. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция для закрытия этого окна. |
| setUrlFile | Функция (UseState), для хранения URL-ссылки на файл в поле urlFile. |

Таблица П17 - Описание методов и параметров файла HomePage.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| HomePage | Реализует окно, в котором показаны обзор приложения, поясняющее видео и информация о разработчиках. |

Таблица П18 - Описание методов и параметров файла InputOption.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| InputOption | Реализует объект Icon-Title в виде кнопки. |
| Параметры | |
| Имя | Назначение |
| Icon | Иконка объекта. |
| title | Заголовок объекта. |
| color | Цвет объекта. |

Таблица П19 - Описание методов и параметров файла JoinPage.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| JoinPage | Реализует страницу, откуда пользователь может вступить в группу. |

Таблица П20 - Описание методов и параметров файла PostItem.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| PostItem | Реализует блок публикаций трех типов: объявление обязательное и необязательное задание. |
| Параметры | |
| Имя | Назначение |
| name | Название канала, где создали эту публикацию. |
| urlPic | URL-ссылка на картину канала, где создали эту публикацию. |
| time | Время публикации. |
| message | Текстовое содержание публикации. |
| deadline | Окончательный срок сдачи обязательной работы. |
| admin | Email человека, который сделал публикацию. |
| postType | Тип публикации (объявление обязательное и необязательное задание). |
| urlFile | URL-ссылка на прикрепленный файл с публикацией. |
| postId | Идентификатор публикации. |

Таблица П21 - Описание методов и параметров файла SubmitPopUp.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| SubmitPopUp | Реализует окно, где обучающийся загружает работу и отправляет ее рецензенту. |
| Параметры | |
| Имя | Назначение. |
| postId | Идентификатор публикации. |
| onClose | Функция, для закрытия этой страницы. |
| channelIdTask | Идентификатор канала. |

Таблица П22 - Описание методов и параметров файла TaskItem.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| TaskItem | Реализует объект необязательного задания. |
| Параметры | |
| Имя | Назначение |
| channelId | Идентификатор канала, где это задание находится. |
| postId | Идентификатор публикации внутри канала. |

Таблица П23 - Описание методов и параметров файла Tasks.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Tasks | Извлечет данные, связанные с необязательным заданием, из firestore и представляет набор данных в виде TaskItem. |

Таблица П24 - Описание методов и параметров файла AddChannelPopUp.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| AddChannelPopUp | Реализует вспомогательное окно, в котором можно добавить новый канал. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция, для закрытия этой страницы. |

Таблица П25 - Описание методов и параметров файла Bottom.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Bottom | Реализует нижнюю часть панели навигации, где можно добавить новый канал и просмотреть список всех существующих каналов. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция для закрытия вспомогательного окна, в котором можно добавить новый канал. |

Таблица П26 - Описание методов и параметров файла Middle.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Middle | Реализует среднюю часть панели навигации, где есть кнопки, через которые можно посмотреть необязательные и обязательные задания. |
| Параметры | |
| Имя | Назначение |
| onClose | Функция для закрытия панели навигации. |

Таблица П27 - Описание методов и параметров файла SidebarOption.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| SidebarOption | Реализует объект Icon-Title в виде кнопки для панели навигации. |
| Параметры | |
| Имя | Назначение |
| Icone | Иконка объекта. |
| title | Заголовок объекта. |

Таблица П28 - Описание методов и параметров файла SidebarSubject.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| SidebarSubject | Реализует объект в виде кнопки, через которой можно открыть выбранный канал. |
| Параметры | |
| Имя | Назначение |
| id | Идентификатор канала. |
| ocClose | Функция для закрытия панели навигации. |

Таблица П29 - Описание методов и параметров файла StudyGoLogo.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| StudyGoLogo | Реализует логотип приложения. |

Таблица П29 - Описание методов и параметров файла Top.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Top | Реализует верхнюю часть панели навигации, где есть название, картина(аватар) группы и имя текущего пользователя. |

Таблица П30 - Описание методов и параметров файла UserSettingSidebar.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| UserSettingSidebar | Реализует выпадающее окно выхода из аккаунта. |
| Параметры | |
| Имя | Назначение |
| userSettingSidebar | Сигнал открытия этого выпадающего окна. |
| ocClose | Функция для закрытия этого выпадающего окна. |

Таблица П31 - Описание методов и параметров файла Login.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Login | Реализует страницу авторизации. |

Таблица П32 - Описание методов и параметров файла MyAppBar.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| MyAppBar | Реализует панель приложения. |

Таблица П33 - Описание методов и параметров файла Sidebar.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| Sidebar | Реализует выпадающее окно панели навигации. |
| Параметры | |
| Имя | Назначение |
| sidebar | Сигнал открытия панели навигации. |
| ocClose | Функция для закрытия панели навигации. |

Таблица П33 - Описание методов и параметров файла AppSlice.js

|  |  |
| --- | --- |
| Методы | |
| Имя | Назначение |
| AppSlice | Реализует Redux приложение, позволяет нам определять редукторы и генерировать связанные действия. |

# Лист регистрации изменений.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | Новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |