

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**  
**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

СОГЛАСОВАНО

УТВЕРЖДАЮ

Приглашенный преподаватель  
департамента программной инженерии  
факультета компьютерных наук

Академический руководитель образовательной  
программы «Программная инженерия», кандидат  
технических наук

 А. Н. Степанов  
«03» мая 2022 г.

 В. В. Шилов  
«03» мая 2022 г.

**КЛИЕНТСКАЯ ЧАСТЬ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ПОМОЩИ В  
ЗАПОМИНАНИИ ТЕОРИТИЧЕСКОЙ ЧАСТИ ДИСЦИПЛИН**


Пояснительная записка

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.10-01 ПЗ 01-1-ЛУ

Исполнитель:

Студент группы БПИ-206

 / А. С. Сибирцева /

«03» мая 2022 г.

Подп. и дата	
Инв. № дубл.	
Взам. Инв. №	
Подп. и дата	
Инв. № подл.	

УТВЕРЖДЕН

RU.17701729.05.10-01 ПЗ 01-1-ЛУ

**КЛИЕНТСКАЯ ЧАСТЬ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ПОМОЩИ В  
ЗАПОМИНАНИИ ТЕОРИТИЧЕСКОЙ ЧАСТИ ДИСЦИПЛИН**

**Пояснительная записка**

**RU.17701729.05.10-01 ПЗ 01-1**

**Листов 87**

Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

**АННОТАЦИЯ**

Пояснительная записка на «Клиентская часть мобильного приложения для помощи в запоминании теоретической части дисциплин» содержит следующие разделы: «Глоссарий», «Введение», «Назначение разработки», «Технические характеристики», «Технико-экономические показатели».

В разделе «Глоссарий» содержатся определения терминов и понятий, используемых в пояснительной записке.

В разделе «Введение» указано наименование и документ, на основе которого ведется разработка.

В разделе «Назначение разработки» указано функциональное и эксплуатационное назначение программного продукта, а также краткая характеристика области применения приложения.

Раздел «Технические характеристики» содержит следующие подразделы: «Постановка задачи на разработку программы», «Описание архитектуры», «Описание алгоритмов работы», «Описание функционирования программы», где разбирается состав выполняемых функций и описание интерфейса и его работы, «Описание и обоснования выбора метода организации входных и выходных данных», «Описание выбора состава технических и программных средств».

Раздел «Технико-экономические показатели» содержит ориентировочную экономическую эффективность, предполагаемую годовую потребность, экономические преимущества разработки приложения.

Настоящий документ разработан в соответствии с требованиями:

- 1) ГОСТ 19.101-77 Виды программ и программных документов [1];
- 2) ГОСТ 19.102-77 Стадии разработки [2];
- 3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];
- 4) ГОСТ 19.104-78 Основные надписи [4];
- 5) ГОСТ 19.105-78 Общие требования к программным документам [5];
- 6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

7) ГОСТ 19.201-78 Пояснительная записка. Требования к содержанию и оформлению [7].

Изменения к данной Пояснительной записке оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

**СОДЕРЖАНИЕ**

ГЛОССАРИЙ .....	5
1. ВВЕДЕНИЕ .....	6
1.1. Наименование программы.....	6
1.2. Документы, на основании которых ведётся разработка.....	6
2. НАЗНАЧЕНИЕ РАЗРАБОТКИ .....	7
2.1. Функциональное назначение.....	7
2.2. Эксплуатационное назначение.....	7
2.3. Краткая характеристика области применения.....	8
3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ .....	9
3.1. Постановка задачи на разработку программы.....	9
3.2. Описание архитектуры .....	9
3.3. Описание алгоритма работы .....	10
3.4. Описание функционирования программы.....	11
3.4.1. Состав выполняемых функций.....	11
3.4.2. Описание интерфейса и его работы .....	11
3.5. Описание и обоснование выбора метода организации входных и выходных данных .....	45
3.5.1. Описание организации входных и выходных данных .....	45
3.5.2. Обоснование выбора метода организации входных и выходных данных.....	45
3.6. Описание и обоснование выбора состава технических и программных средств .....	45
3.6.1. Состав технических и программных средств .....	45
3.6.2. Обоснование выбора технических и программных средств .....	46
4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ .....	47
4.1. Ориентировочная экономическая ценность .....	47
4.2. Предполагаемая потребность .....	47
4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами .....	47
СПИСОК ИСТОЧНИКОВ .....	48
ПРИЛОЖЕНИЕ 1.....	49
ПРИЛОЖЕНИЕ 2.....	53
ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ.....	87

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## ГЛОССАРИЙ

- 1) *Пользователь* – человек, который использует приложение «Stoady». В контексте приложения «Stoady» существует два типа пользователей: «студенты» и «администраторы».
- 2) *Студент* (также, *участник, участник команды*) – пользователь, цель которого – лучше разобраться в теоретическом материале дисциплин.
- 3) *Администратор* – пользователь, имеющий доступ к редактированию материалов, содержащихся в приложении «Stoady», а именно:
  - добавление, удаление и редактирование прав участников команд;
  - добавление, удаление и редактирование предметов, тем и вопросов, содержащихся в команде.
- 4) *Аватар* – изображение, которое используется в качестве персонального изображения пользователя или команды.
- 5) *Команда* (также, *группа*) - группа людей, объединенных общим набором предметов.
- 6) *Предмет* (также, *дисциплина*) – верхний уровень иерархии теоретических материалов в приложении, который может содержать внутри себя одну или несколько тем.
- 7) *Тема* – раздел предмета, содержащий необходимую для изучения теорию и соответствующие ей вопросы.
- 8) *Вопрос* – элемент нижнего уровня иерархии теории в приложении. Вопрос содержит в себе сам текст вопроса, а также ответ на него.
- 9) *API (Application Programming Interface)* – описание способов (набор классов, методов и т. п.), которыми одна компьютерная программа (в данном случае, клиентская часть приложения) может взаимодействовать с другой (в данном случае, с сервером).
- 10) *Snackbar* – элемент, который позволяет выводить всплывающие сообщения, растягивая их по ширине экрана.
- 11) *Виджет* – элемент графического интерфейса, выполняющий како-то действие

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

**1. ВВЕДЕНИЕ****1.1. Наименование программы**

Наименование программы – «Клиентская часть мобильного приложения для помощи в запоминании теоретической части дисциплин».

Наименование программы на английском языке – «Client Side of the Mobile Application for Assisting in the Memorization of Theoretical Parts of Disciplines».

Краткое наименование программы для пользователя: «Stoady»

**1.2. Документы, на основании которых ведётся разработка**

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 "Программная инженерия" и утвержденная академическим руководителем тема курсового проекта.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 2. НАЗНАЧЕНИЕ РАЗРАБОТКИ

### 2.1. Функциональное назначение

Мобильное приложение «Stoady» представляет собой сервис для помощи учащимся вузов и других учебных заведений в подготовке к различным элементам контроля. Сервис реализует в себе функциональность, позволяющую пользователям выбирать нужный предмет, а также тему для изучения внутри этого предмета.

Каждый пользователь может создавать команды и приглашать в них других пользователей, чтобы предоставить им доступ к материалам, содержащимся в их команде. Это позволит учащимся адаптировать приложение под образовательную программу их учебного заведения, что позволит лучше ее усвоить: у учащихся не будет необходимости в поиске и фильтрации информации из сети Интернет – вся теория, нужная им, будет сосредоточена в одной команде.

Клиентская часть мобильного приложения предназначена для взаимодействия пользователя с серверной частью и данными посредством графического интерфейса.

### 2.2. Эксплуатационное назначение

Мобильное приложение «Stoady» будет применяться в сфере обучения. Инициативные студенты (а также преподаватели учебных заведений) могут в доступном формате давать необходимую теорию своим сокурсникам (или же подопечным студентам), которая будет агрегирована в одном приложении, что избавит студентов от необходимости хранить все ресурсы и ссылки на них в каких-либо хранилищах (например, Google Drive). Это улучшит пользовательский опыт студентов при подготовке к элементам контроля, а наличие функциональности для отслеживания статистики при изучении теории будет способствовать более эффективному обучению – каждый студент будет понимать, какие темы ему стоит повторить, на каких вопросах следует заострить внимание.

Целевая аудитория приложения – студенты, которые хотят лучше изучить теоретическую часть дисциплин или помочь своим сокурсникам сделать это благодаря добавлению необходимой теории в свою команду.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



### 2.3. Краткая характеристика области применения

Целевая аудитория мобильного приложения «Stoady» – студенты, которым требуется помощь в запоминании теоретической части учебных дисциплин (к примеру, формулировок, теорем, билетов к экзамену и т. п.). Студенты смогут выбирать необходимый для изучения предмет, проходить тесты по темам, изученным в курсе данной дисциплины, а также отслеживать свою статистику и выявлять слабые места в изученной по выбранному предмету теории.

Помимо этого, пользователям будет доступна возможность создавать команды, чтобы готовиться к экзаменам и другим элементам контроля совместно со своими друзьями или сокурсниками. Приложение также могут использовать преподаватели, публикуя теорию и сопутствующие вопросы по дисциплинам для студентов своих групп.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

### 3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

#### 3.1. Постановка задачи на разработку программы

Разрабатываемая программа должна соответствовать описанным в техническом задании функциональным характеристикам, требованиям к интерфейсу и надежности, а также временным характеристикам.

#### 3.2. Описание архитектуры

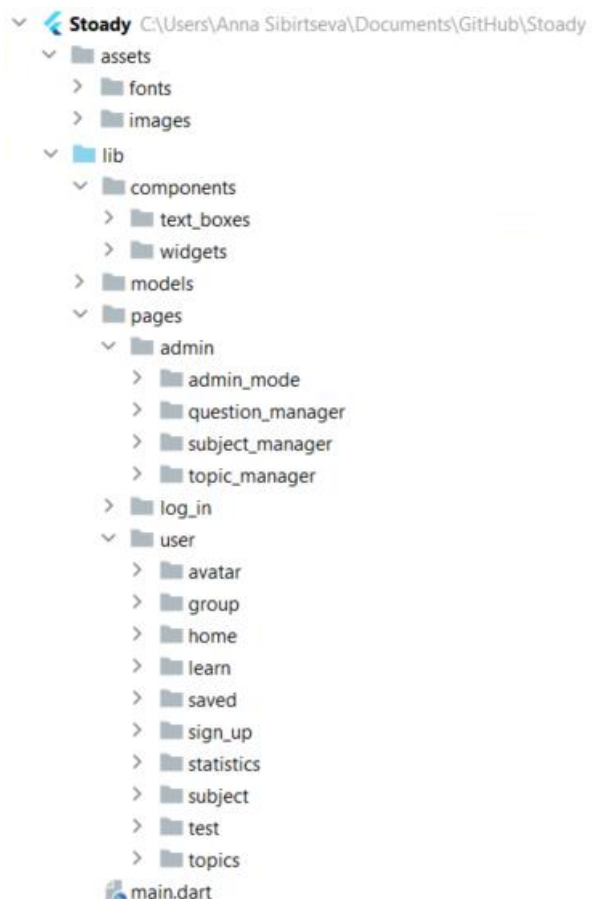


Рисунок 1. Структура проекта.

На рисунке 1 представлена архитектура проекта. В папке “assets” хранятся шрифты (папка “fonts”) приложения, а также необходимые изображения, такие как разные иконки, логотип и т. д. (папка “images”).

В папке “components” хранятся все созданные дополнительно виджеты: поля для ввода, кнопки, меню и т. д.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

В папке “models” можно найти описание всех классов и сущностей, используемых в приложении (подробнее смотрите в приложении 1 «Описание и функциональное назначение классов» и приложении 2 «Описание и функциональное назначение полей, методов и свойств»).

Папка “pages” содержит в себе код описания для всех экранов, где каждый выделен в отдельную папку с одноименным названием. Структура каждой папки-страницы представлена на рисунке 2 на примере экрана авторизации. У каждого экрана есть основной файл с кодом, где происходит его создание, а также папка “components”, в которой хранятся описания переднего и заднего плана. Чаще всего под задним планом подразумеваются элементы интерфейса, не имеющие смысловой нагрузки и созданы только ради дизайна, например: лейблы, изображение и т. д. Передний план в свою очередь отвечает за все элементы интерфейса, с которыми пользователь может как-то взаимодействовать. Данная архитектура помогает разбивать экраны на слои, для упрощения работы с ними.

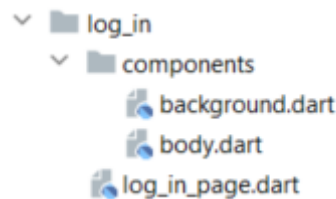


Рисунок 2. Структура папок с созданием экранов.

### 3.3. Описание алгоритма работы

Обмен данными между серверной и клиентской частями осуществляется посредством HTTP- методов. Было использовано 4 вида HTTP-запросов: POST, PUT, GET и DELETE:

- HTTP-запрос POST был применен для отправки новых данных на сервер: для создания нового предмета, темы, вопроса, группы, пользователя, добавления его в группу. Так же для разного рода управления объектами, например: редактирования предметов, аватара команды, темы и т. д.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- HTTP- запрос PUT был применен для обновления существующей информации на сервере. Для редактирования информации о пользователе, его сохраненных вопросов, и его роли в како-либо команде.
- HTTP- запрос GET был применен для получения данных с сервера о ранее созданных объектах: пользователь и информации о нём, сохраненные вопросы, группы, в которых он состоит и его роли в них, статистика и т. д.
- HTTP- запрос DELETE был применен для удаления существующих данных на сервере, например: предметов, тем, вопросов, пользователей из команд.

### 3.4. Описание функционирования программы

#### 3.4.1. Состав выполняемых функций

Список всех выполняемых функций можно найти в настоящем Техническом задании «Мобильное приложение для помощи в запоминании теоретической части дисциплин».

#### 3.4.2. Описание интерфейса и его работы

Для удобства «Описание интерфейса и его работы» имеет сокращенную нумерацию: ссылаясь на разделы, содержащиеся в «Описание интерфейса и его работы», из других разделов или других документов, стоит использовать расширенную нумерацию, т. е. пункт «1. Авторизация» стоит нумеровать, как «3.4.2.1. Авторизация».

Так же описание запросов к серверу будет иметь следующий формат:

<ТИП ЗАПРОСА>: <КОНЕЧНАЯ ТОЧКА API>

### 1. Авторизация

При открытии приложения первый экран – экран авторизации (Рисунок 3). Он содержит следующие элементы, для входа в аккаунт:

- 1.1. Поле для ввода почты.
- 1.2. Поле для ввода пароля, которое автоматически скрывает введенный пароль.
- 1.3. Кнопка, для подтверждения введенных данных и входа в аккаунт («LOG IN»).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1.4. Кнопка для перехода на экран регистрации (подробнее в разделе «2. Регистрация»), обозначенная соответствующим текстом «SIGN UP»

После ввода данных и нажатия на кнопку, описанную в пункте 1.3 выше, осуществляется запрос к серверу посредством POST: /auth/authorize запроса, в который мы передаем введенную ранее почту и пароль.

Если ответом сервера был код ошибки 401, то программа понимает, что пользователь не был авторизован и на экране появляется соответствующее сообщение (Рисунок 4)

Если же ответ сервера – 200, значит введенные данные были найдены в базе данных и введены корректно. В этом случае пользователь успешно входит в приложение и его перенаправляют на экран его команд (смотрите пункт «3. Команды»)

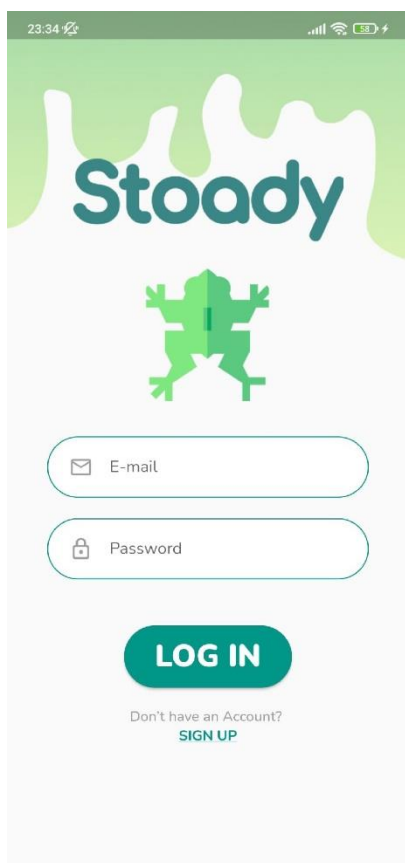


Рисунок 3. Экран Авторизации

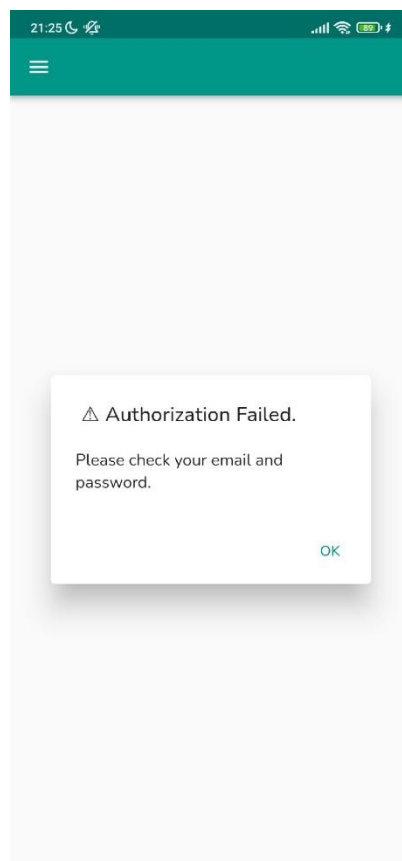


Рисунок 4. Ошибка Авторизации

## 2. Регистрация

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Для регистрации нового пользователя необходимо нажать на соответствующий элемент дизайна на экране авторизации (смотрите пункт 1.4 в разделе «1. Авторизация»).

Экран Регистрации содержит следующие элементы:

- 2.1. Поле ввода имени.
- 2.2. Поле ввода почты.
- 2.3. Поле для ввода пароля, которое автоматически скрывает введенный пароль.
- 2.4. Кнопка для подтверждения введенных данных и подтверждения регистрации («SIGN UP»).
- 2.5. Кнопка для перехода на экран авторизации (подробнее в разделе «1. Авторизация»), обозначенная соответствующим текстом «SIGN IN».

После заполнения всех полей и нажатия на кнопку, описанную в пункте 2.4 выше, происходит первичная проверка на валидность введенных данных. Если поля были заполнены некорректно, то пользователю показывается соответствующее сообщение в формате `snackbar`'а (Рисунок 6, Рисунок 7, Рисунок 8). Для удобства полноценный экран показан только на Рисунке 6, на остальных же показано только сообщение, которое появляется аналогично). Так, например

- Имя считается корректным, если в нем содержится от 2 до 30 символов.
- Почта считается корректной, если в ней содержится от 7 до 100 символов.
- Пароль считается корректным, если содержит от 8 до 30 символов.

Если все данные были введены корректно, то программа сначала генерирует аватар пользователя (смотрите раздел 12. Выбор аватара для подробной информации), а потом осуществляет `POST: /users/register` запрос к серверу, передавая ему введенные данные и новый аватар.

Если код, вернувшийся с сервера – 400, то регистрация не могла быть осуществлена, так как пользователь с введенной почтой уже существует в базе данных.

Если же сервер вернул код 200, то регистрация успешно завершается и пользователя направляют на экран его команд (смотрите раздел «3. Команды»), а также программа получает сгенерированный `id` для пользователя, для дальнейшей работы.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

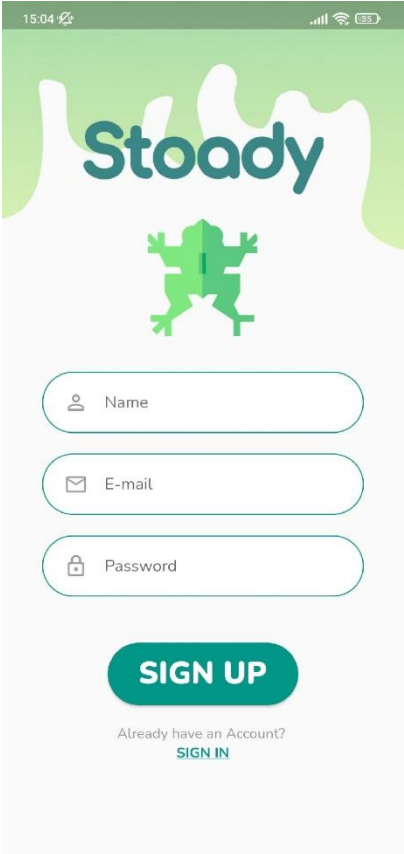


Рисунок 5. Экран Регистрации

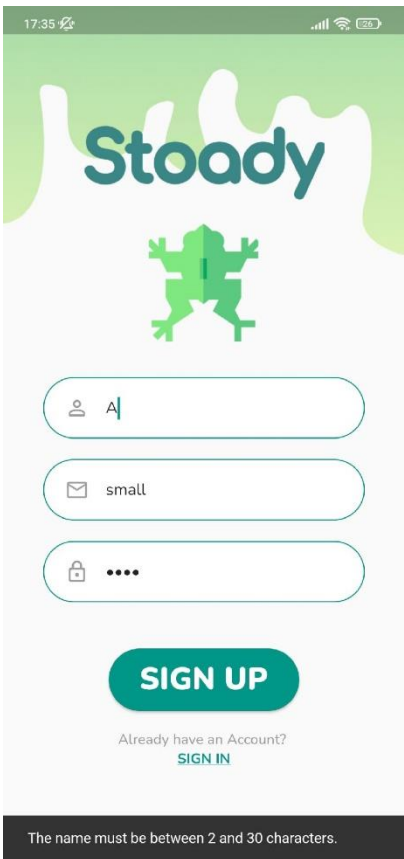


Рисунок 6. Сообщение о некорректно  
заполненном поле "Name"

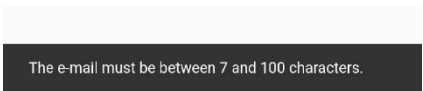


Рисунок 7. Сообщение об ошибке при  
некорректном заполнении поля "E-mail"



Рисунок 8. Сообщение об ошибке при  
некорректном заполнении поля  
"Password"

3. Команды

После успешной авторизации/регистрации пользователь попадает на экран со всеми, командами, в которых он состоит. Программа получает эти данные при помощи GET: /users/teams запроса к серверу, передавая при этом идентификатор пользователя.

Экран команды состоит из следующих элементов:

- 3.1. Лейбл, показывающий, что пользователь сейчас находится на экране со своими командами.
- 3.2. Список команд, где каждый элемент представляется в виде названия и аватара команды. Если у пользователя нет команд, то список отображается пустым (Рисунок 9, Рисунок 10)
- 3.3. Текстовое поле, для названия новой команды.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

3.4. Кнопка для подтверждения создания новой команды с введенным в текстовом поле, описанным в пункте 3.3., названием.

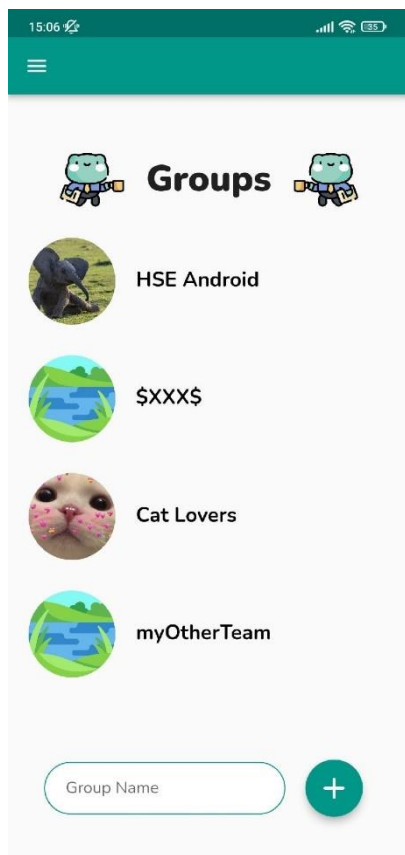


Рисунок 9. Экран Команд, если список групп пользователя не пустой

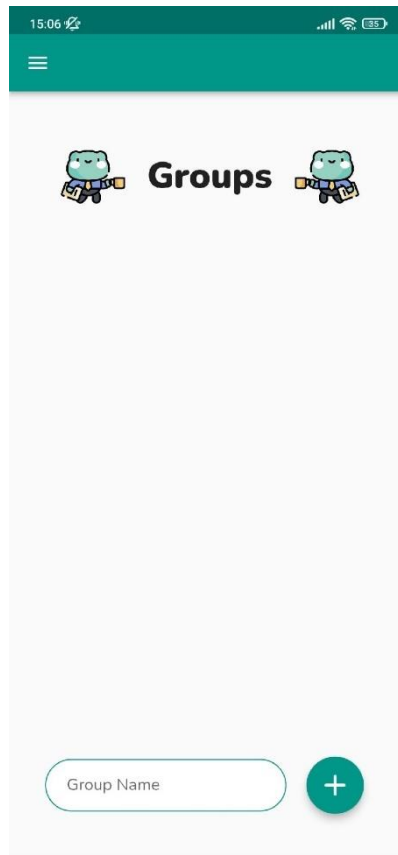


Рисунок 10. Экран Команд, если список групп пользователя пуст

При нажатии на кнопку, описанную выше в пункте 3.4, программа отправляет на сервер POST: /teams/create запрос с идентификатором (id) пользователя, который теперь будет создателем команды (смотрите пункт 13. Режим администратора для подробной информации) и названием команды. В случае, если пользователь не указал имя, оно инициализируется автоматически значением “New Team”. Так же всем новым группам присваивается дефолтный аватар (Рисунок 11)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата





Рисунок 11. Стандартный аватар созданной команды

Так же пользователь может перейти в любую группу, которая отображена в списке, просто нажав на нее.

При выполнении этого действия программа отправляет на сервер GET: /teams/{teamId} запрос, передавая идентификатор выбранной команды и получая подробную информацию о ней, а именно: название, изображение и список предметов, соответствующих этой группе. Эта информация затем отображается на экране, на который перенаправляется пользователь (смотрите раздел «4. Список предметов команды»)

#### 4. Список предметов команды

При нажатии на выбранную из списка команду пользователь попадает на экран с информацией о ней со следующими элементами:

- 4.1. Лейбл с названием команды (если его длина больше 11 символов, то остальные обрезаются и вместо них показывается многоточие - ..., рисунок 14)
- 4.2. Список предметов, представленный в виде названия и иконки кувшинки (Рисунок 12, а если в группе нет предметов, то данный список будет пустым: Рисунок 13).

Пользователь может перейти на экран с информацией о предмете путём нажатия на выбранный элемент списка. При выполнении этого действия программа посылает серверу GET: /subjects/{subjectId} запрос, передавая идентификатор выбранной дисциплины и получая информацию о ней, а именно: название, описание и список прикрепленных к ней тем.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

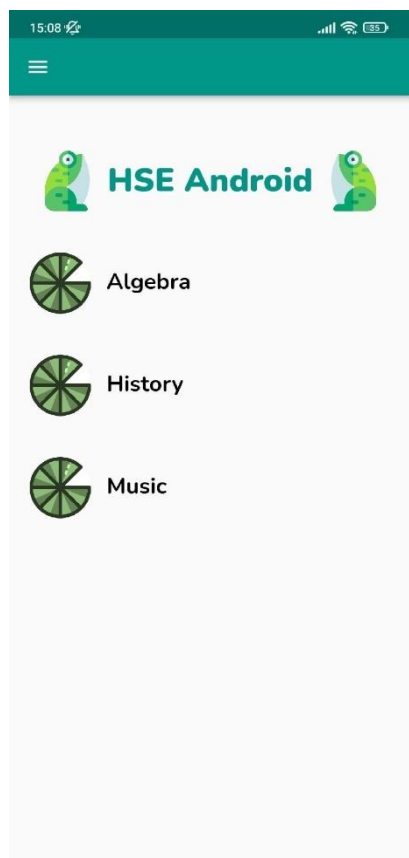


Рисунок 12. Экран со списком предметов команды, если он не пустой.

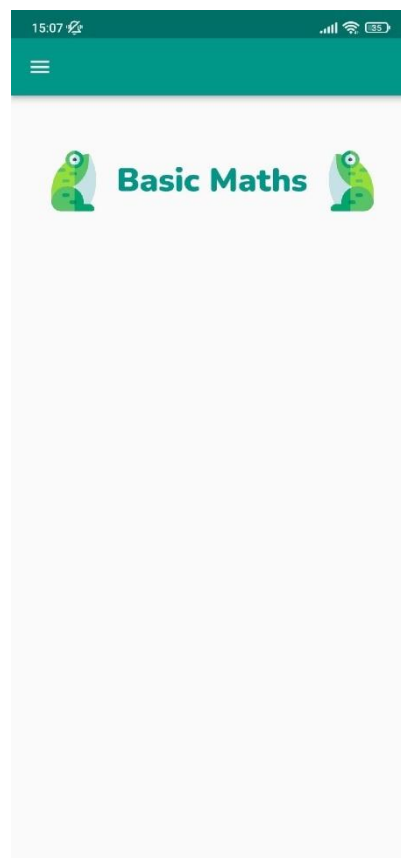


Рисунок 13. Экран со списком предметов команды, если он пуст.



Рисунок 14. Формат отображения названия команды в случае, если длина имени больше 11 символов.

## 5. Предметы

При нажатии на выбранный элемент списка с предметами пользователь попадает на экран с информацией о выбранном элементе со следующими компонентами:

5.1. Лейбл с названием выбранного предмета

5.2. Описание предмета

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

5.3. Список тем, соответствующих выбранной теме (Рисунок 15, если таковых нет, то список отображается пустым Рисунок 16)

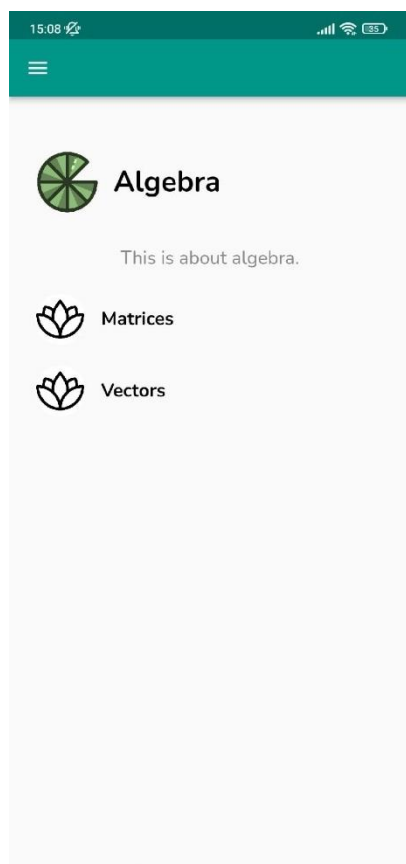


Рисунок 15. Экран с информацией о предмете, если в нем есть темы

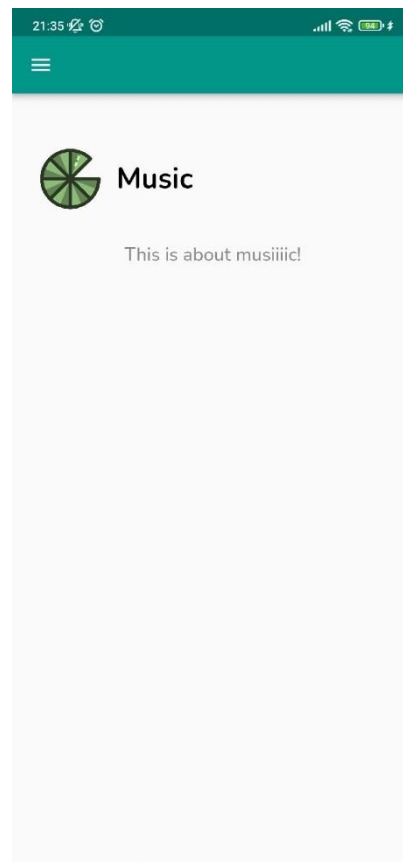


Рисунок 16. Экран с информацией о предмете, если в нем нет тем.

При нажатии на любую существующую тему программа отправляет на сервер GET: `/topics/{topicId}` запрос, передавая идентификатор выбранной темы и получая необходимую информацию: имя, описание и список вопросов. После сохранения всех полученных данных пользователь попадает на страницу предмета.

## 6. Темы

На экране предмета пользователь может видеть следующие элементы интерфейса:

- 6.1. Название предмета.
- 6.2. Описание предмета.
- 6.3. Кнопка, позволяющая изучать вопросы, прикрепленные к теме. (“Learn”)
- 6.4. Кнопка, позволяющая проходить тестирование по прикрепленным к теме вопросам. (“Test”)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

При нажатии на кнопку, описанную выше в пункте 6.3 или пункте 6.4, приложение

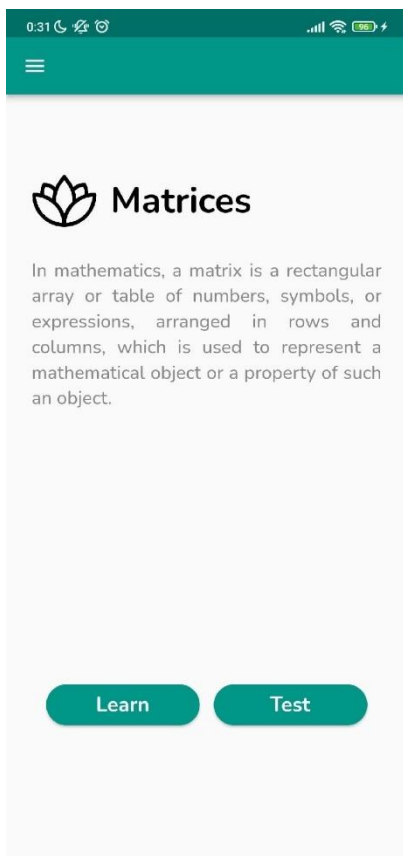


Рисунок 17. Экран темы.

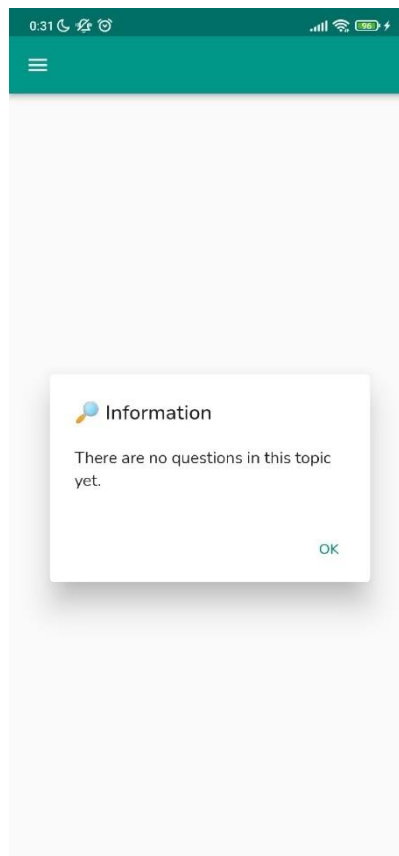


Рисунок 18. Информация об ошибке, если в теме нет вопросов.

отправляет GET: /questions/{topicId} запрос, передавая идентификационный номер темы, которую выбрал пользователь.

В ответ, если вернувшийся код был 200, программа получает данные в формате списка вопросов, относящихся к теме, и перенаправляет пользователя на экран изучения (смотрите раздел 7. Изучение) или тестирования (смотрите раздел 8. Тестирование) в зависимости от того, на какую из кнопок он нажал.

Если этот список оказывается пустым, то пользователю показывается диалоговое окно, оповещающее его о том, что ему пока что нечего учить. (Рисунок 18)

## 7. Изучение

На экране с изучением темы пользователь увидит следующие элементы интерфейса:

7.1. Лейбл с названием предмета, к которому относится вопрос.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- 7.2. Лейбл с названием темы, к которой относится вопрос.
- 7.3. Виджет в форме пятиконечной звезды для добавления текущего вопроса в сохраненные (смотрите раздел 10. Сохраненные вопросы).
- 7.4. Интерактивная карточка с текстом вопроса и ответа.
- 7.5. Счетчик, показывающий номер текущего вопроса и общее количество вопросов в теме.
- 7.6. Кнопки-стрелки для переключения между вопросами.

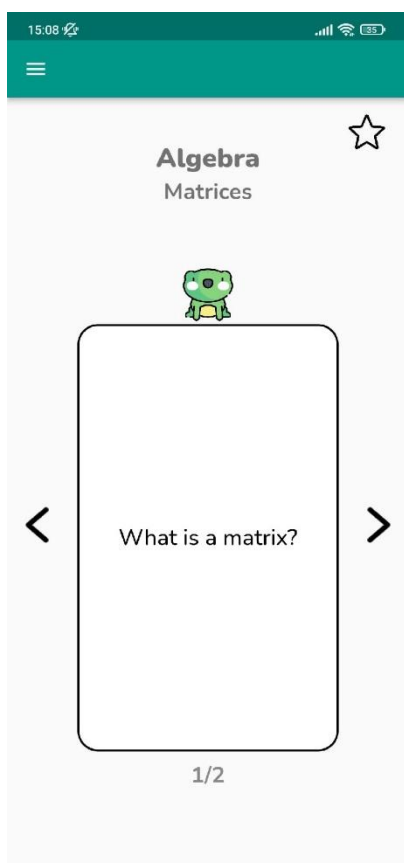


Рисунок 19. Экран обучения, если текущий вопрос не сохранен.

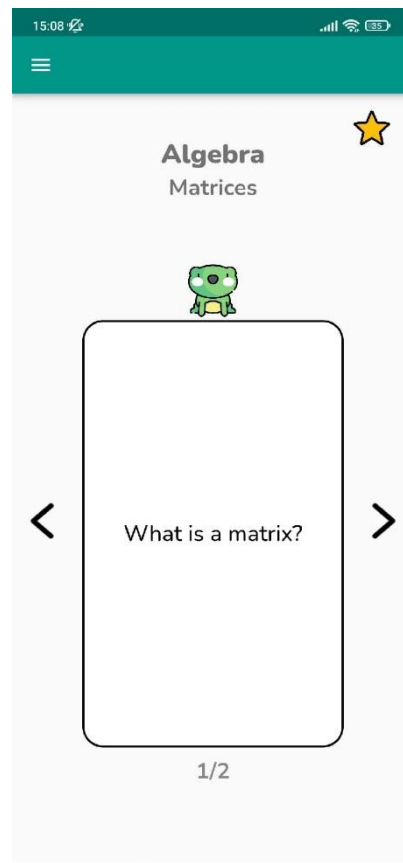


Рисунок 20. Экран обучения, если текущий вопрос сохранен.

Работа с карточками, описанными в пункте 7.4 происходит следующим образом: изначально она показывает текст вопроса. При нажатии в любую ее область (до черных границ) она переворачивается и текст вопроса заменяется ответом.

Пользователь может переключаться между вопросами при помощи кнопок-стрелок, описанных в пункте 7.6. Стрелка влево возвращает пользователя на предыдущий вопрос, а вправо – следующий. В случае, если счетчик становится равным 1 и пользователь

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

пытается пролистать назад (стрелка влево), то ничего не происходит. Аналогично работает и для случая, когда номер вопроса –  $n$  и пользователь нажимает на стрелку вправо.

При нажатии на виджет, описанный в пункте 7.3., программа проверяет, был ли вопрос сохранен до этого, если нет (звезда была пустой, Рисунок 19), то она отправляет на сервер PUT: /questions/save/{questionId} запрос, чтобы сохранить его. Если же вопрос до этого уже находился в сохраненных (звезда была закрашена, Рисунок 20), то программа отправляет на сервер DELETE: /questions/unsave/{questionId} запрос, чтобы удалить его из списка. В обоих случаях, если вернувшийся код равен 200, то программа перерисовывает виджет на противоположный, т. е. если звезда была закрашена, то она становится пустой и аналогично в обратную сторону. Если же программа получила другой код, то она оповещает пользователя о том, что действие (удаление или добавление) не было выполнено при помощи snackbar'a (подробнее смотрите в разделе 10. Сохраненные вопросы).

## 8. Тестирование

Экран тестирования очень похож на экран изучения (смотрите пункт 7. Изучение). Он содержит абсолютно все те же элементы интерфейса и логику, за исключением интерактивных карточек. На данном экране при нажатии на них ничего не происходит, т. е. ответ не показывается.

Так же данный экран содержит два других элемента:

8.1. Текстовое поле для ввода ответа (Рисунок 21)

8.2. Кнопка, подтверждающая введенный текст и отправку ответа на проверку.  
(Рисунок 22)

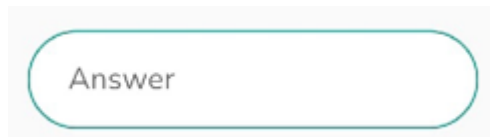


Рисунок 21. Текстовое поле для ввода ответа.

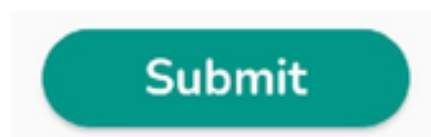


Рисунок 22. Кнопка для отправки введенного ответа.

После заполнения текстового поля, описанного в пункте 8.1. и нажатия на кнопку, описанную в пункте 8.2 происходит первичная проверка: введенный текст отправится на

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

проверку только при условии, что в ячейке было что-то написано. Поэтому, если пользователь просто нажал на кнопку “Submit” при этом не написав ничего в поле для

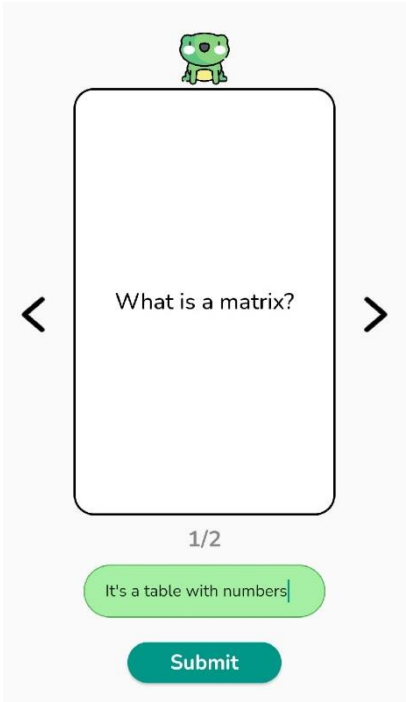


Рисунок 23. Экран тестирования в случае, если ответ пользователя был верным.

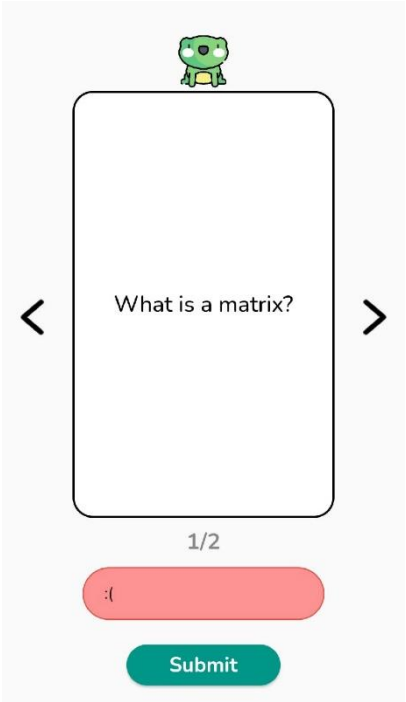


Рисунок 24. Экран тестирования в случае, если ответ пользователя был не верным.

ответа, то ничего не произойдет. Если же первичная проверка была пройдена, то программа сверяет введенный ответ с тем, что хранится в списке по текущему номеру вопроса. В случае, если то, что ввел пользователь полностью совпадает с ответом, то ячейка подсвечивается зеленым (Рисунок 23), в противном случае – красным (Рисунок 24). Отвечать на вопрос даётся неограниченное количество раз.

В любой момент пользователь может закончить тестирование просто покинув экран. В этот момент программа посчитает его результат поделив количество верных ответов на общее количество вопросов в тесте и сохранит в статистике (смотрите раздел 9. Статистика), отправив POST: /tests запрос, передавая итоговые результаты.

9. Статистика

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Так же пользователю доступен просмотр статистики по пройденным тестам. Попасть на данный экран можно из бокового меню (смотрите раздел 11. Боковое меню). На экране с результатами находятся следующие элементы интерфейса:

- 9.1. Лейбл, показывающий пользователю, что он находится на экране статистики.
- 9.2. Список результатов, представленный в виде названия темы и иконки короны, а также процентная доля верных ответов (если пользователь не проходил тестов, то данный список будет пустым).

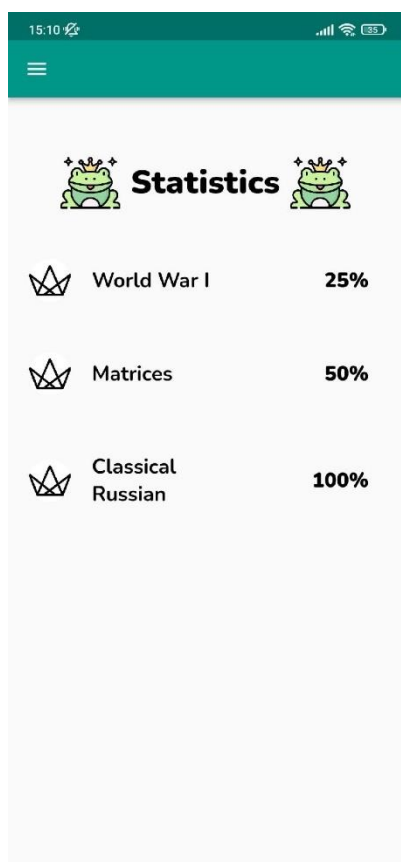


Рисунок 25. Экран статистики.

Процент для каждой темы в отдельности рассчитывается по формуле:  $N_{\text{выполненных}} / N_{\text{вопросов}} * 100\%$ , где  $N_{\text{выполненных}}$  – число верных ответов на вопросы в данной теме при последнем прохождении тестирования по этой теме,  $N_{\text{вопросов}}$  – число вопросов в данной теме.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Процент отображается как округленное до целого числа, по правилам математики (если дробная часть составляет от 0 включительно до 0.5 не включительно, то процент округляется вниз; если же она составляет от 0.5 включительно до 1 не включительно, то процент округляется вверх)

## 10. Сохраненные вопросы

Экран сохраненных вопросов представляет собой упрощенную версию экрана предметов (смотрите раздел 6. Предметы). В него можно попасть при помощи бокового меню (смотрите раздел 11. Боковое меню). На нем представлены следующие элементы:

- 10.1. Лейбл, показывающий пользователю, что он находится на экране сохраненных вопросов (“Saved Questions”).
- 10.2. Кнопка, позволяющая изучать сохраненные вопросы (“Learn”).
- 10.3. Кнопка, позволяющая проходить тестирование по прикрепленным к теме вопросам (“Test”).

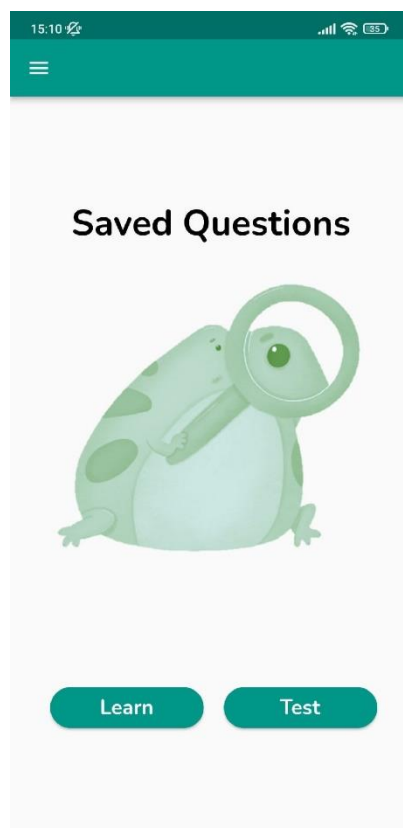


Рисунок 26. Экран сохраненных вопросов.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Аналогично экрану с предметами (смотрите раздел 6. Предметы) при нажатии на кнопки, описанные выше в пунктах 10.3 и 10.4 программа отправляет на сервер GET: /questions/saved/{userId} запрос передавая идентификатор пользователя и получая при этом список его сохраненных вопросов. а затем, в зависимости от того куда он нажал, перенаправляет его на экраны тестирования или изучения, описанные в разделах: 7. Изучение, 8. Тестирование.

## 11. Боковое меню

Для упрощения навигации, в приложении есть боковое меню, которое пользователь может открыть, находясь на любом из возможных экранов кроме авторизации и регистрации. Меню состоит из следующих элементов:

- 11.1. Аватар пользователя (смотрите раздел 12. Выбор аватара)
- 11.2. Имя пользователя
- 11.3. Почта пользователя
- 11.4. Раздел сохраненных вопросов
- 11.5. Раздел статистики
- 11.6. Раздел команд
- 11.7. Раздел предметов
- 11.8. Раздел режима администратора (смотрите раздел 13. Режим администратора)
- 11.9. Кнопка выхода из аккаунта

Элементы перечислены в порядке своего отображения в меню (Рисунок 27, Рисунок 28).

При нажатии на аватар, описанный в пункте 11.1, пользователь попадает на экран выбора аватара (подробнее в разделе 12. Выбор аватара)

Так же каждый раздел перенаправляет пользователя на соответствующие страницы, так, например, при нажатии на:

- 11.4 пользователь попадет на экран с сохраненными вопросами (смотрите раздел 10. Сохраненные вопросы)
- 11.5 пользователь попадает на экран, где будет отображаться его статистика (смотрите раздел 9. Статистика)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- 11.6 пользователь попадет на экран со всеми своими командами (смотрите раздел 3. Команды)
- 11.7 пользователь попадет на экран со всеми предметами в выбранной группе. В случае, если пользователь не выбрал никакую группу и пытается перейти на экран со списком предметов в группе, то ему отображается экран с ошибкой (Рисунок 29)
- 11.9 пользователь попадает на экран авторизации, где может сменить аккаунт или зарегистрировать новый (смотрите раздел 1. Авторизация)

Изначально раздел режима администратора, описанный в пункте 11.8 является не нажимаемым, пока пользователь не перейдет в группу, в которой является администратором. Тогда его перенаправят на экран управления командой (смотрите раздел 13. Режим администратора и рисунки 27 и 28).

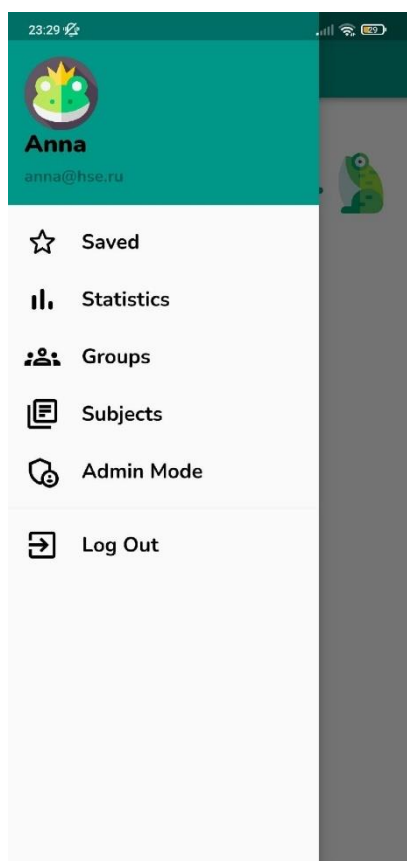


Рисунок 27. Боковое меню, если режим администратора доступен

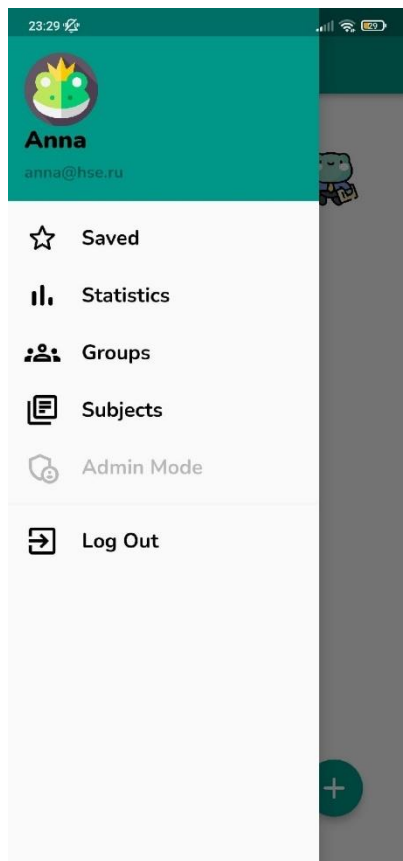


Рисунок 28. Боковое меню, если режим администратора заблокирован.

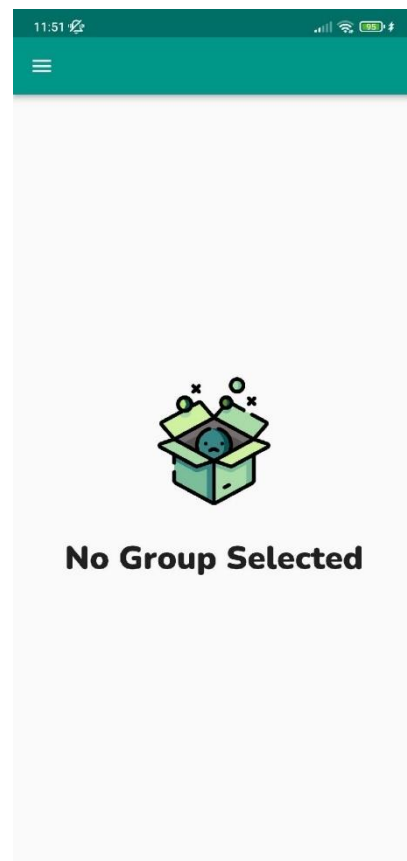


Рисунок 29. Экран ошибки в случае, если пользователь не выбрал группу и пытается перейти в предметы.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 12. Выбор аватара

Как уже упоминалось в разделе 2. Регистрация, при регистрации пользователю присваивается случайно сгенерированный аватар из пятнадцати возможных, представленных на рисунке 30 ниже.



Рисунок 30. Все доступные пользователю аватары.

При желании пользователь может сменить фото своего профиля на другое благодаря экрану выбора аватара, который открывается через боковое меню, путём нажатия на изображение, как уже упоминалось выше в разделе 11. Боковое меню.

На экране выбора аватара пользователь может увидеть следующие элементы интерфейса:

12.1. Лейбл-подсказка для пользователя, с текстом (“Select Avatar”).

12.2. 15 различных возможных картинок на выбор.

Среди представленных изображений, выбранный аватар единственный отображается не замутненный. Для изменения картинки, пользователю нужно просто нажать на нее. Тогда выбранный элемент станет выделяться на фоне остальных из-за непрозрачности (смотрите рисунки 31 и 32).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Рисунок 31. Экран выбора аватара, если текущий аватар находится в нижнем правом углу.

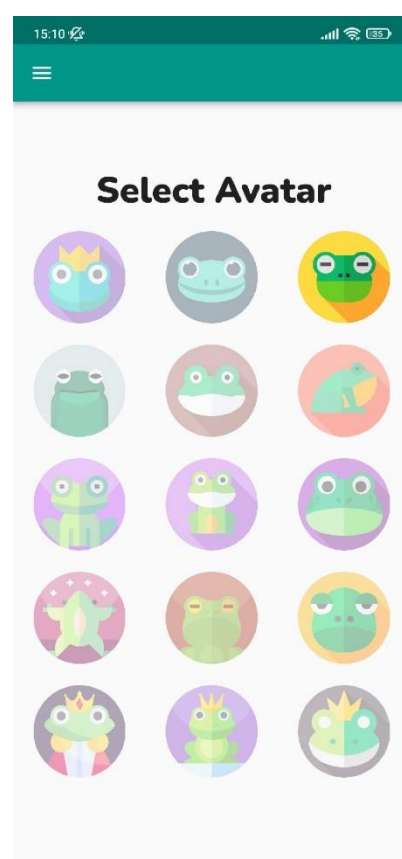


Рисунок 32. Экран выбора аватара, если текущий аватар находится в верхнем правом углу.

При выполнении данного действия на сервер будет отправлен PUT: /users/{userId}/avatar/set запрос, в который мы передаем идентификатор пользователя и индекс нового аватара

### 13. Режим администратора

Администратором группы считается человек, который либо является создателем группы, либо был назначен им в качестве администратора.

Если у пользователя есть права администратора для выбранной группы, то ему доступен текущий раздел, в котором он может управлять различными аспектами команды.

На экране управления командой пользователь может увидеть следующие элементы интерфейса:

#### 13.1. Лейбл с названием текущей команды.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

13.2. Кнопка дополнительной информации (Рисунок 33)

13.3. Текстовое поля для почты нового участника команды.

13.4. Кнопка для подтверждения добавления нового пользователя с введенной в текстовом поле, описанным в пункте 13.2., почтой.

13.5. Список участников группы, где каждый элемент списка представлен в виде его почты, которая выделена жирным шрифтом в случае, если пользователь является создателем и переключателя, отвечающего за его роль в команде (В положении «вправо» (и выделенный цветом) – участник имеет права администратора. В положении «влево» (не выделенный цветом) – человек имеет прав администратора, он просто участник команды). (Рисунки 34 и 35)

13.6. Кнопки для фильтрации участников (Рисунки 36 и 37)

13.7. Кнопка для управления предметами текущей команды.



Рисунок 33. Кнопка дополнительной информации.



Рисунок 34. Переключатель, если участник не обладает правами администратора.



Рисунок 35. Переключатель, если участник обладает правами администратора.

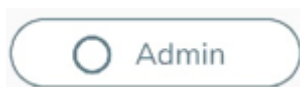


Рисунок 36. Кнопка для фильтрации студентов не активированная.



Рисунок 37. Кнопка для фильтрации студентов активированная.

Полностью экран режима администратора представлен на рисунке 38. При нажатии на кнопку дополнительной информации, изображенную на рисунке 33, пользователю показывается небольшое диалоговое окно с информацией об управлении командой, а именно: инструкция по тому, как менять название и аватар группы, а так же, как удалять участников из группы (Рисунок 39).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

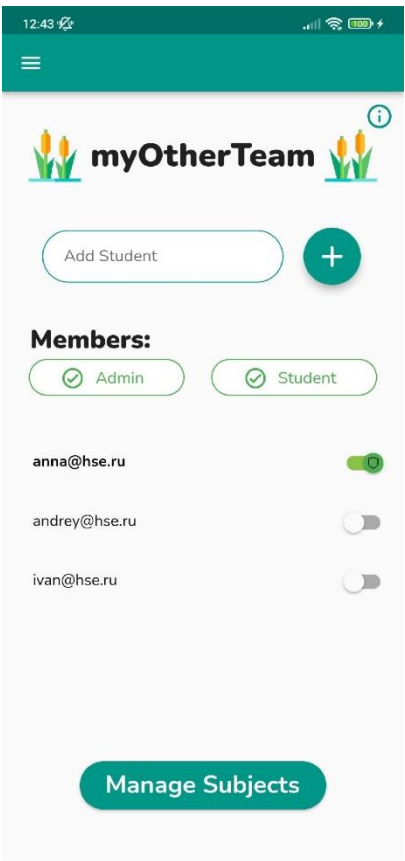


Рисунок 38. Экран управления командой (режима администратора).

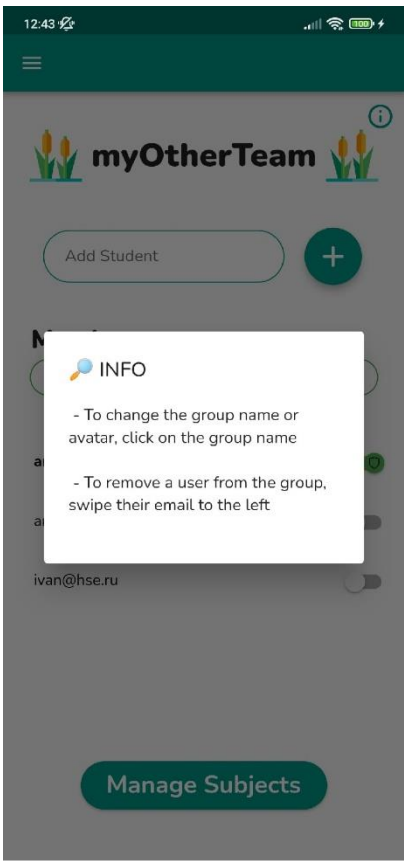


Рисунок 39. Информационное окно.

При нажатии на лейбл с названием группы, описанный в пункте 13.1 пользователю открывается окно настроек, в котором он может поменять название или аватар группы, где аватар представляется в формате прямой ссылки на изображение. При некорректном формате предоставленной ссылки изображение отображаться не будет. У окна так же присутствуют две кнопки: “CANCEL” и “SAVE”. При нажатии на первую, внесённые изменения не сохраняются, а окно просто закрывается, в то время как при нажатии на вторую, на сервер будет отправлен POST: teams/{teamId}/avatar запрос, передавая идентификатор текущей группы и новые параметры для команды, чтобы сохранить внесенные изменения. (Рисунок 40)

Администратор может добавлять новых пользователей по их почте. Для этого ему надо заполнить соответствующее текстовое поле, описанное в пункте 13.3 и после подтверждения данного действия путем нажатия на кнопку, описанную в пункте 13.4 на

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

сервер, будет отправлен POST: /teams/{teamId}/members/add запрос. Передавая идентификатор команды и пользователя, а также введенную почту, команда ожидает ответа. Если полученный код был равен 200, то программа понимает, что пользователь был успешно добавлен в команду и обновляет список участников. В противном случае пользователь не был добавлен, так как введенная почта была некорректная и пользователь видит соответствующее сообщение в формате snackbar'a (Рисунок 41)

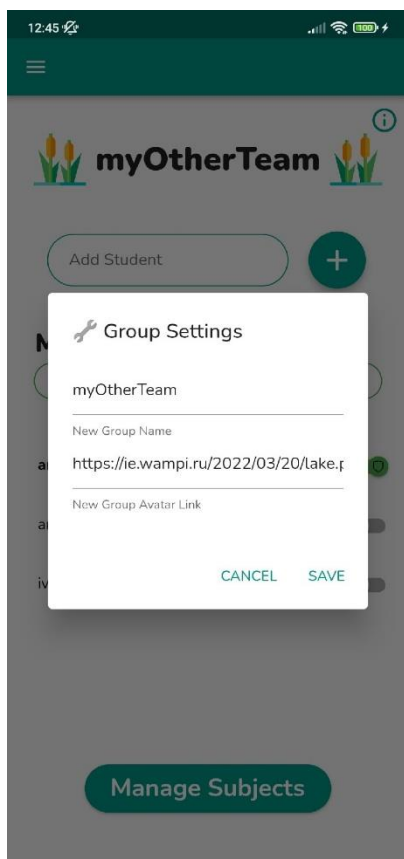


Рисунок 40. Окно настроек названия и аватара команды.

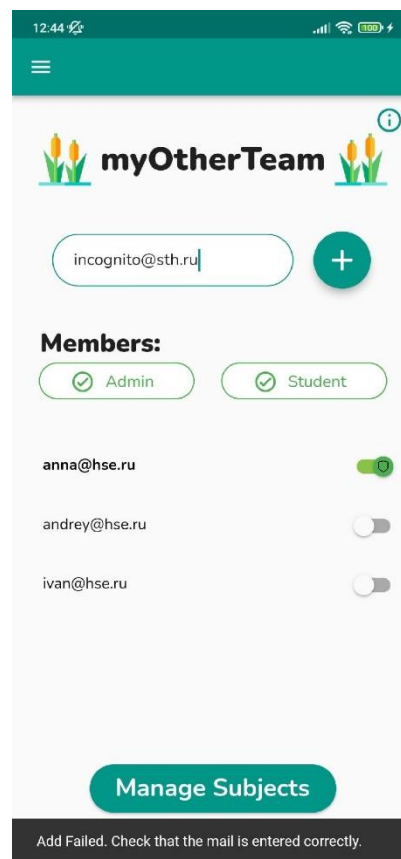


Рисунок 41. Сообщение об ошибке в случае, если пользователь не был добавлен в команду.

Как уже писалось выше, администратор может фильтровать список участников, отображаемый на экране при помощи кнопок, описанных в пункте 13.6. При переходе на экран оба параметра и администраторов, и студентов стоят включёнными и подсвечивается зеленым, а в кружочке отображается галочка (Рисунок 37). При нажатии на данную кнопку в списке участников не будут показываться те, у кого была данная роль, а сама кнопка окрасится в серый цвет, и галочка пропадет (Рисунок 36). Например, при нажатии на кнопку администратора из списка пропадут все участники, которые обладают

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



данной ролью (Рисунок 42) и аналогично в обратную сторону при нажатии на кнопку “Student”. Если оба параметра фильтрации будут выключены, то список будет пустым.

Администратор может менять роли участников при помощи переключателя,

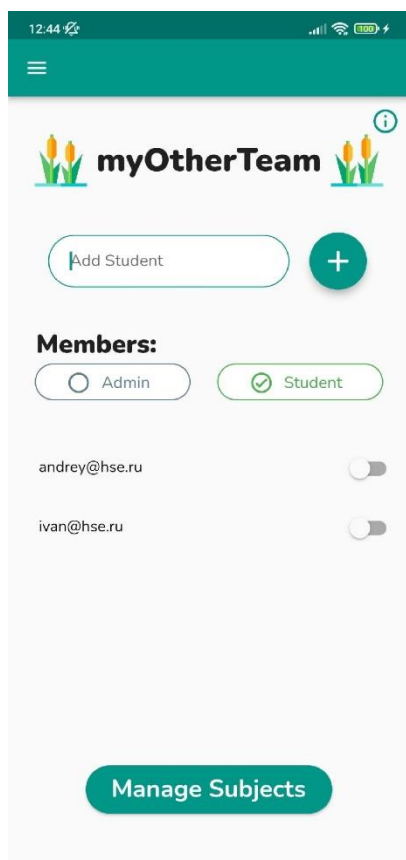


Рисунок 42. Список участников команды без администраторов.

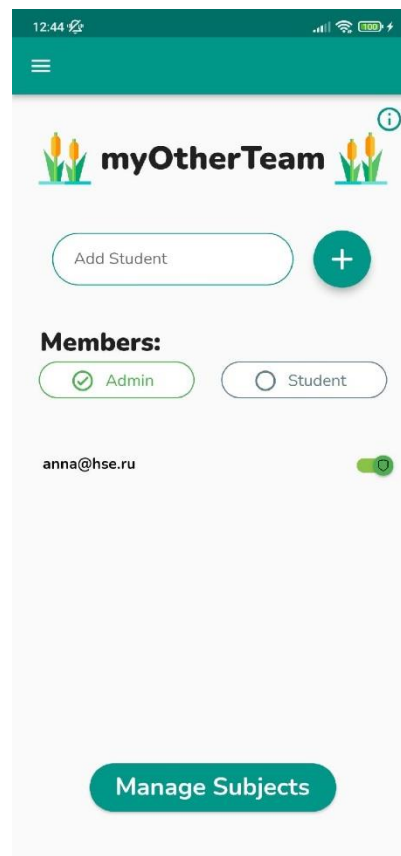


Рисунок 43. Список участников команды только из администраторов.

описанного в пункте 13.5. При выполнении этого действия происходит первичная проверка действия. Программа проверяет, что пользователь, чью роль меняют, не является создателем, так как менять его роль нельзя. В случае, если кто-то пытается это сделать, он получает соответствующее сообщение в формате snackbar’а (Рисунок 44). Если же первичная проверка была пройдена, то программа отправляет на сервер PUT: /teams/{teamId}/members запрос, передавая идентификатор текущей команды, пользователя, выполняющего действие и пользователя, чью роль надо поменять и на какую. Новая роль пользователя определяется программой, как обратная к текущей. То есть если пользователь обладал правами администратора, он становится обычным

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

участников группы и теперь не сможет заходить в режим администратора. Аналогично работает и в обратную сторону.

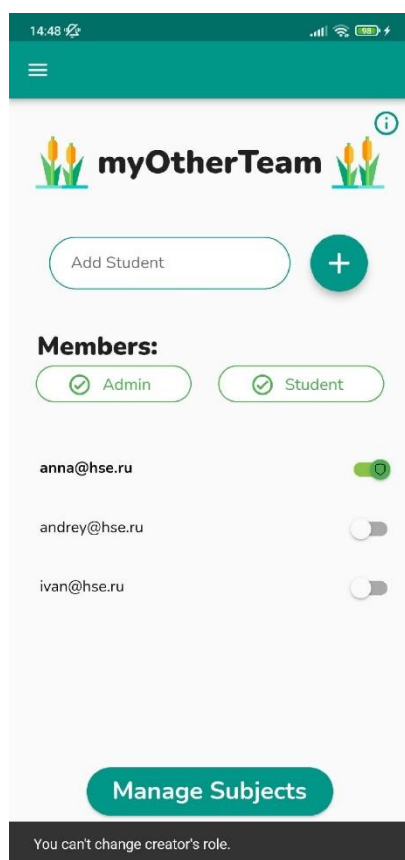


Рисунок 44. Ошибка при попытке изменения роли создателя.

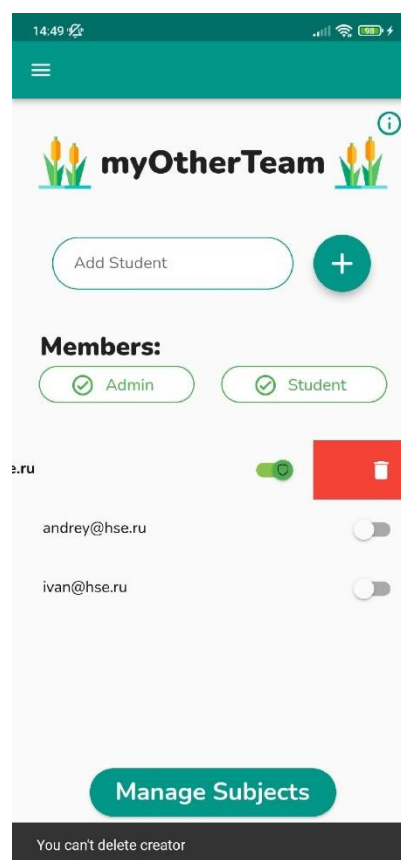


Рисунок 45. Ошибка при попытке удаления создателя.

Так же администратор может удалять пользователей из группы путем проведения пальцем влево карточки с информацией о пользователе. При данном действии первичная проверка так же смотрит, чтобы удаляемым пользователем, был не создатель команды, так как его удалять нельзя. Если же кто-то пытается удалить создателя, он получает соответствующее сообщение в формате snackbar'a (Рисунок 45). В случае, если первичная проверка пройдена, то пользователю показывается диалоговое окно (Рисунок 46), в котором он может либо подтвердить, либо отменить удаление. Если он нажимает "АССЕПТ", то на сервер отправляется DELETE: /teams/{teamId}/members/remove запрос, с идентификаторами пользователя, выполняющего действие, пользователя, которого надо удалить и команды, из которой выбранный участник будет убран. В случае, если

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

полученный с сервера код будет 200, то программа понимает, что удаление прошло успешно и оповещает пользователя о том, что выбранный им участник был удален из команды (Рисунок 47). Если же код не равен 200, то программа оповещает администратора, что удаление не было выполнено из-за ошибки.

При нажатии на кнопку, описанную выше в пункте 13.7 администратор переходит на экран управления предметами (смотрите раздел 14. Управление предметами).

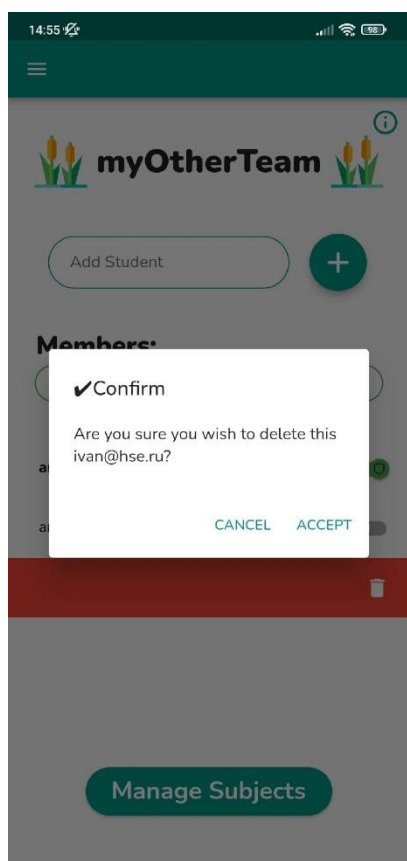


Рисунок 46. Диалоговое окно для подтверждения удаления пользователя.

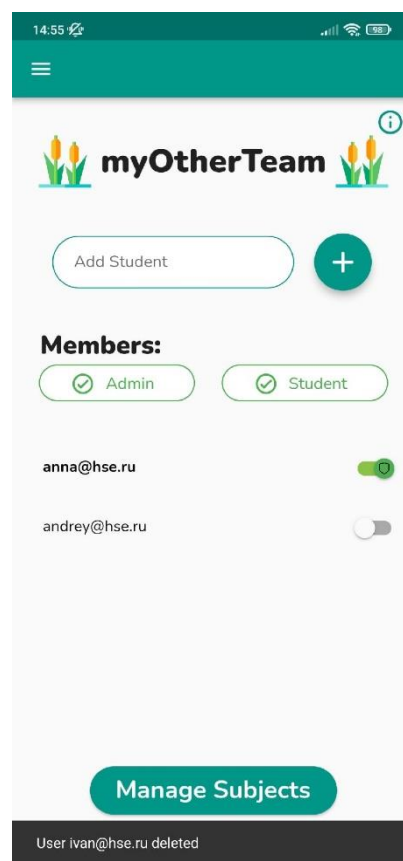


Рисунок 47. Сообщение о том, что пользователь был успешно удален.

## 14. Управление предметами

Экран управления предметами является копией экрана со списком предметов команды (смотрите пункт 4. Список предметов команды), но с небольшими изменениями. Он содержит в себе те же элементы интерфейса, но помимо них на нем так же присутствует кнопка добавления нового предмета в группу. При нажатии на нее

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

пользователь попадает на экран редактирования предмета (подробнее смотрите в разделе 15. Редактирование предметов).

А также основным отличием данного экрана от экрана со списком предметов является то, что при нажатии на элементы списка пользователь попадает не на экран с описанием предмета, описанный в пункте 5. Предметы, а на экран редактирования этого предмета (подробнее в разделе 15. Редактирование предметов).

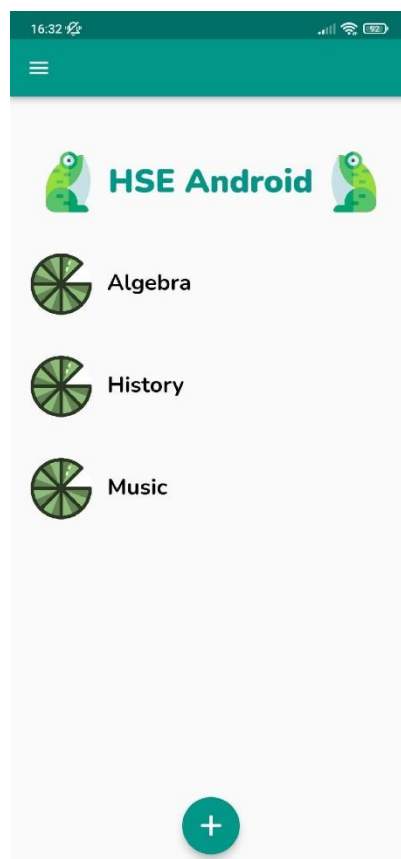


Рисунок 48. Экран редактирования предметов.

## 15. Редактирование предметов

Экран редактирование предмета состоит из следующих элементов интерфейса:

- 15.1. Лейбл, показывающий пользователю, что он сейчас находится на экране редактирования предметов.
- 15.2. Кнопка с иконкой мусорной корзины для удаления предмета (Рисунок 49).
- 15.3. Кнопка для добавления новой темы к предмету (Рисунок 50).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- 15.4. Текстовое поле для имени предмета, в которое нельзя ввести больше 50 СИМВОЛОВ.
- 15.5. Текстовое поле для описания предмета, в которое нельзя ввести больше 250 СИМВОЛОВ.
- 15.6. Список всех тем, принадлежащих данному предмету, где каждый элемент представлен названием темы. Если у предмета нет тем, то данный список отображается пустым.
- 15.7. Кнопка отмены всех изменений (“Cancel”).
- 15.8. Кнопка сохранения изменений (“Save”).



Рисунок 49.  
Кнопка удаления.



Рисунок 50.  
Кнопка  
добавления.

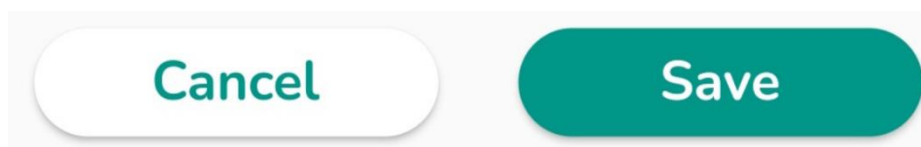


Рисунок 51. Кнопки отмены и сохранения изменений.

На данный экран можно попасть двумя способами, как было написано выше. В случае, если пользователь нажал на кнопку добавления предмета, описанную в предыдущем разделе, то текстовые поля, описанные в пунктах 15.4 и 15.5 будут пустыми, за исключением текстов-подсказок в которых написано, для чего нужно данное текстовое поле. (Рисунок 52)

В случае, если пользователь попыбует нажать на кнопку добавление предмета, описанную в пункте 15.3 предварительно не сохранив новый предмет, он увидит диалоговое окно с ошибкой. (Рисунок 53)

Так же, если пользователь попыбует удалить предмет, которые еще не был сохранен, он увидит окно с ошибкой, оповещающее его, что так как предмет, который он сейчас рассматривает, не был сохранен, программе нечего удалять (Рисунок 54). Если же предмет уже существует, то пользователю показывается диалоговое окно, в котором он

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

может либо подтвердить удаление, либо отменить его. В случае, когда он подтверждает удаление предмета, программа отправляет на сервер DELETE: /subjects/remove/{subjectId} запрос, передавая идентификатор предмета, который нужно удалить. В случае, если вернувшийся код не равен 200, то пользователю выводится сообщение, что что-то пошло не так и предмет не был удален. Если же вернувшийся код был 200, то удаление было успешно завершено и пользователь возвращается на экран управления предметами.

Второй способ попасть на данный экран – нажать на элемент списка, о котором говорилось ранее в разделе 14. Управление предметами. При выполнении данного действия экран выглядит аналогично, за исключением того, что текстовые поля, описанные в пунктах 15.4 и 15.5 выше, не будут пустыми. Они будут заполнены текущими значениями, соответствующими данному предмету (Рисунок 55).



Рисунок 52. Экран редактирования предметов при создании нового предмета.

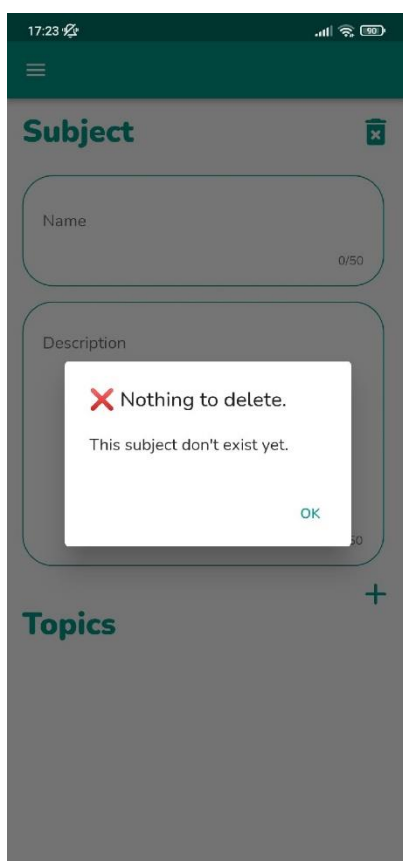


Рисунок 53. Окно с ошибкой при попытке удаления несохраненной темы.

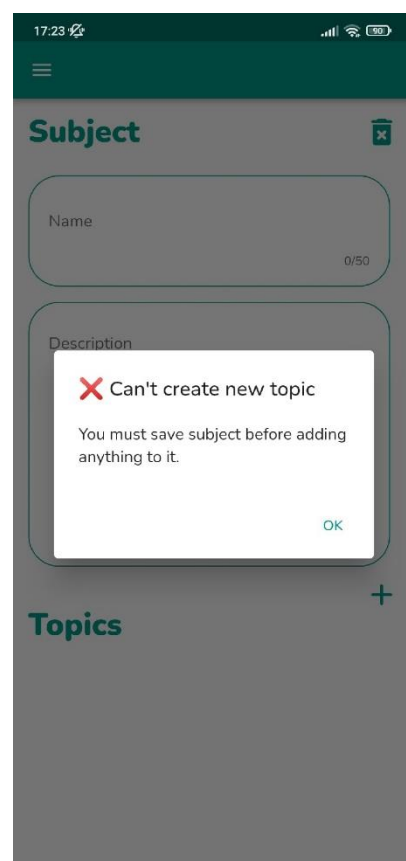


Рисунок 54. Окно с ошибкой при попытке добавить темы в несуществующий предмет.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

При нажатии на кнопку “Cancel”, описанную в пункте 15.7, все изменения не сохраняются, и пользователь возвращается на экран управления предметами (смотрите раздел 14. Управление предметами).

При нажатии на кнопку “Save”, описанную в пункте 15.8, происходит первичная проверка, которая смотрит, чтобы поле имени обязательно было заполнено, если же данное поле оказывается пустым, то пользователь видит окно с соответствующей ошибкой (Рисунок 56). Если первичная проверка проходит успешно, то программа смотрит: создал ли пользователь новый предмет или просто отредактировал существующий. Если пользователь хочет сохранить измененные данные для существующего предмета, то программа отправляет на сервер POST: /subjects/edit/{subjectId} запрос, передавая идентификатор текущего предмета и новые значения для описания и имени предмета.

Если же пользователь создал новый предмет, то программа отправляет на сервер POST: /subjects/add запрос, передавая при этом идентификатор текущей группы, в которую теперь будет добавлен этот предмет, а также значения текстовых полей для имени и писания предмета.

Если предмет, который редактировал администратор, был сохранен в базе данных, то он может добавить в него новую тему, посредством нажатия на кнопку, описанную в пункте 15.3. В этом случае он попадет на экран редактирования тем. (подробнее в разделе 16. Редактирование тем).

Так же пользователь может перейти к редактированию существующего предмета посредством нажатия на элемент списка тем принадлежащих выбранному предмету, описанного в пункте 15.6.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Рисунок 55. Экран редактирования предмета при открытии существующего предмета.

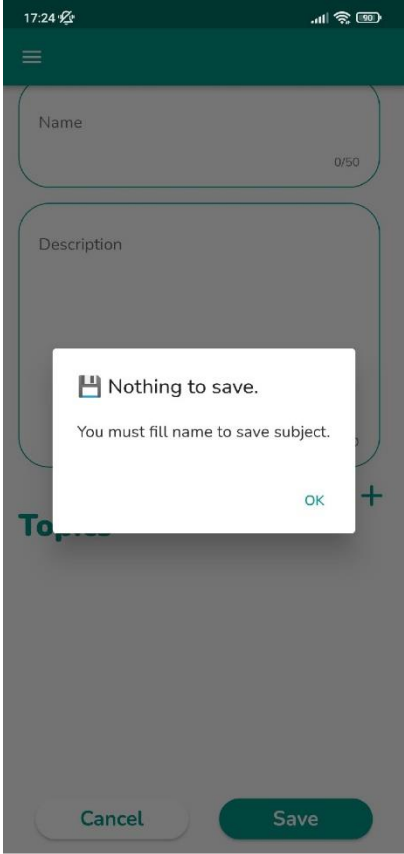


Рисунок 56. Окно с ошибкой при попытке сохранить предмет без имени.

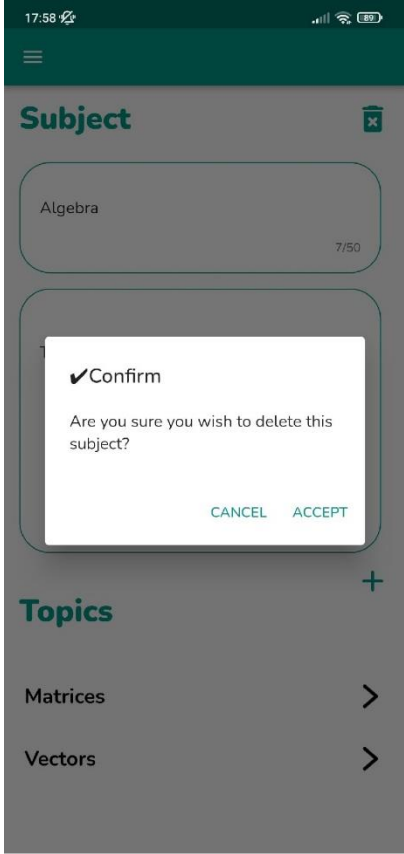


Рисунок 57. Экран подтверждения удаления предмета.

16. Редактирование тем

Экран редактирования тем очень похож по дизайну на экран редактирования предметов. Он так же содержит следующие элементы интерфейса:

- 16.1. Лейбл, показывающий пользователю, что он сейчас находится на экране редактирования тем.
- 16.2. Кнопка с иконкой мусорной корзины для удаления темы. (Рисунок 49).
- 16.3. Кнопка для добавления нового вопроса к теме (Рисунок 50).
- 16.4. Текстовое поле для названия темы, в которое нельзя ввести больше 50 символов.
- 16.5. Текстовое поле для описания темы, в которое нельзя ввести больше 250 символов.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



16.6. Список вопросов, принадлежащих теме, где каждый элемент списка представляется в формате текста вопроса. Если в теме нет вопросов, то данный список отображается пустым.

16.7. Кнопка отмены всех изменений (“Cancel”).

16.8. Кнопка сохранений всех изменений (“Save”).

На данном экране происходят аналогичные с экраном редактирования предметов проверки. Единственно отличие, что текст ошибок изменяется и вместо “subject” пишется “topic”.

Рисунок 58. Экран редактирования тем, если пользователь создает новую тему.

Рисунок 59. Экран редактирования тем в случае, если тема уже существовала.

Конечно же так же меняются запросы к серверу при выполнении действий. Так, например удаление теперь происходит по запросу DELETE: /topics/edit/{topicId}, добавление новой темы через запрос POST: /topics/add, куда, как и в добавлении предмета передаются название и описание, только вместо идентификатора группы теперь передается идентификатор предмета, которому принадлежит данная тема. И наконец запрос на редактирование темы осуществляется через POST: /topics/edit/{topicId}.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Ниже представлены текст ошибок, при попытке сделать тоже, что приводило к ошибкам в работе с предметами (смотрите раздел 16. Редактирование тем).

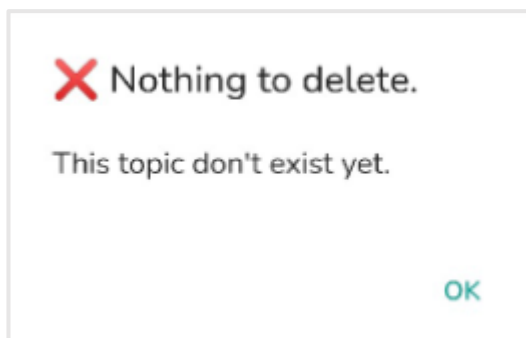


Рисунок 60. Окно ошибки при попытке удаления несуществующей темы.

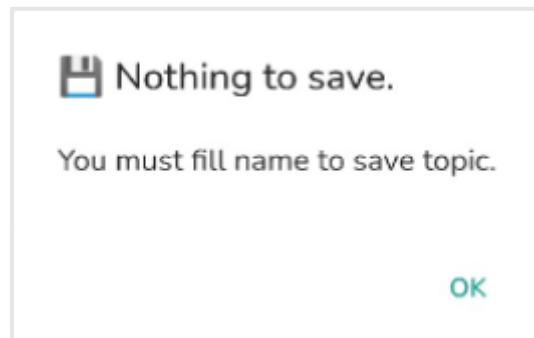


Рисунок 61. Окно ошибки при попытке сохранения темы без имени.

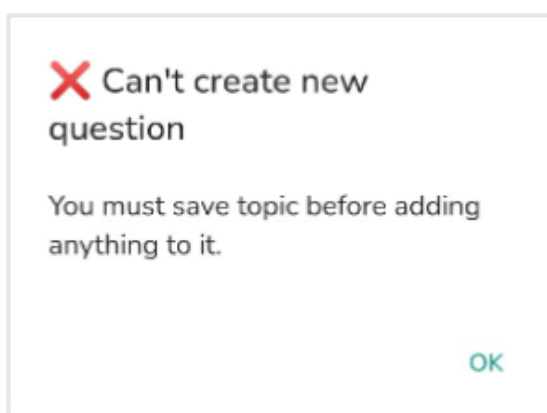


Рисунок 62. Окно ошибки при попытке создать вопрос в несохраненной теме.

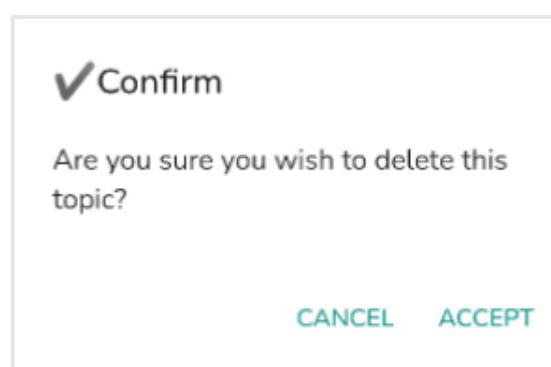


Рисунок 63. Окно подтверждения удаления темы.

Если тема, которую редактировал администратор, была сохранена в базе данных, то он может добавить в нее новый вопрос, посредством нажатия на кнопку, описанную в пункте 16.3. В этом случае он попадет на экран редактирования вопросов. (подробнее в разделе 17. Редактирование вопросов).

Так же пользователь может перейти к редактированию существующего вопроса посредством нажатия на элемент из списка вопросов, принадлежащих выбранной теме, описанного в пункте 16.6.

## 17. Редактирование вопросов

На экране редактирование вопросов можно найти следующие элементы интерфейса:

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

- 17.1. Лейбл, показывающий пользователю, что он сейчас находится на экране редактирования вопросов.
- 17.2. Кнопка с иконкой мусорной корзины для удаления темы. (Рисунок 47).
- 17.3. Текстовое поле для вопроса, в которое нельзя ввести больше 100 символов.
- 17.4. Текстовое поле для ответа, в которое нельзя ввести больше 100 символов.
- 17.5. Кнопка отмены всех изменений (“Cancel”).
- 17.6. Кнопка сохранений всех изменений (“Save”).

Рисунок 64. Экран редактирования вопросов, при создании нового вопроса.

Рисунок 65. Экран редактирования вопросов, при редактировании существующего вопроса.

Если пользователь попал на экран редактирования вопросов путем нажатия на кнопку добавления нового вопроса, описанную в предыдущем разделе в пункте 16.3, то ячейки для вопроса и ответа будут пустыми, а на их месте будет текст-подсказка, для чего они предназначены (Рисунок 64).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Если же пользователь попал на данный экран, потому что нажал на один из вопросов, представленных в списке вопросов, принадлежащем теме и описанном в пункте 16.6 в предыдущем разделе, то оба поля будут уже заполнены ранее введенными значениями (Рисунок 65).

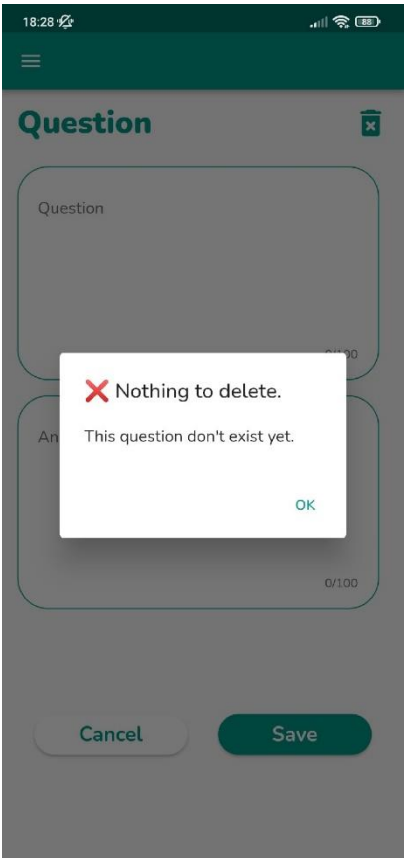


Рисунок 66. Окно с ошибкой при попытке удаления несуществующего вопроса.



Рисунок 67. Окно ошибки при попытке сохранения вопроса с хотя бы одним незаполненным полем

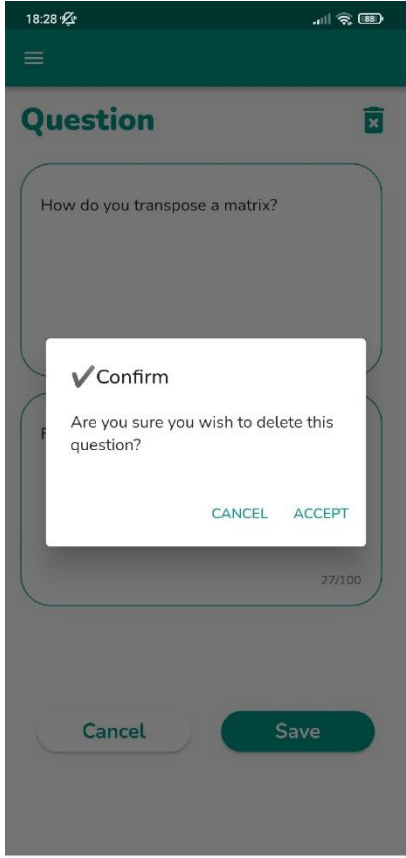


Рисунок 68. Окно подтверждения удаления вопроса.

При нажатии на кнопку сохранения, описанную в пункте 17.6 происходит первичная проверка, в ходе которой программа прослеживает, чтобы оба текстовых поля, описанных в пунктах 17.1 и 17.2 были не пусты. Если же хотя бы одно из них не содержит значения, то пользователь видит окно с ошибкой (Рисунок 66). Если же первичная проверка была пройдена успешно, то программа смотрит как пользователь попал на экран. Если он создавал новый вопрос, то на сервер отправляется POST: /questions/add запрос, в который передаются значение, заданные вопросу и ответу, а также идентификационный номер темы, в которую данный вопрос будет добавлен. Если же пользователь редактировал

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

существующий вопрос, то программа отправляет на сервер POST: /questions/edit/{questionId}, передавая идентификационный номер вопроса, в который будут вноситься изменения и новые параметры для значений вопроса и ответа.

При нажатии кнопки удаления вопроса, описанной в пункте 17.2 появляется окно подтверждения удаления, в котором пользователь либо соглашается с удалением, либо нажимает “CANCEL” и возвращается на предыдущий экран. Так же, происходит первичная проверка, которая удостоверяется в том, что вопрос существовал до этого, если же это не так, то пользователь видит окно с соответствующей ошибкой (Рисунок 67). Если первичная проверка была пройдена успешно, то на сервер посылается DELETE: /questions/remove/{questionId} запрос, в который мы передаем идентификационный номер вопроса, который нужна удалить и текущей темы. Если удаление прошло успешно и сервер вернул код 200, то мы обновляем список вопросов, иначе мы оповещаем пользователя, что удаление не было выполнено.

Если пользователь нажимает на кнопку “Cancel”, описанную в пункте 17.5, то все изменения не сохраняются, а он возвращается на экран редактирования тем.

## 18. Экран ошибки

В случае, если у пользователя нет подключения к интернету на любом из этапов пользования приложением, ему показывается экран ошибки, потому что программа не может корректно функционировать, т. к. практически на каждом этапе нам нужна информация из базы данных, а получить ее без запросов к серверу невозможно. (Рисунок 69)



Рисунок 69. Текст ошибки, если у пользователя нет подключения к Интернету.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

### **3.5. Описание и обоснование выбора метода организации входных и выходных данных**

#### **3.5.1. Описание организации входных и выходных данных**

Входные данные приложения – различные действия пользователя при взаимодействии с интерфейсом, вводимая им информация: строки, числа и прочие данные, а также информация, которую мы получаем посредством HTTP-запросов.

Выходные данные приложения – это отображаем интерфейс и HTTP-запросы серверу.

#### **3.5.2. Обоснование выбора метода организации входных и выходных данных**

Описанные выше входные данные были выбраны так как они являются оптимальным способом получения информации от пользователя для дальнейшей работы.

Благодаря выходным данным в формате интерфейса пользователю будет легче ориентироваться в приложении, а HTTP-запросы способствуют тому, что в приложении всегда будут отображаться новейшие данные.

### **3.6. Описание и обоснование выбора состава технических и программных средств**

#### **3.6.1. Состав технических и программных средств**

##### **3.6.1.1. Состав технических средств**

Для функционирования клиентской части приложения требуется мобильное устройство, оснащенное следующими техническими компонентами:

- 1) Дисплей с минимальным разрешением  $720 \times 1280$  пикселей;
- 2) Операционная система: Android 10 или более поздняя версия Android, или iOS 13 или более поздняя версия iOS;
- 3) Минимальный объем оперативной памяти – 2 Гб;
- 4) Минимальный объем свободной памяти устройства – 500 Мб.
- 5) Сенсорный экран
- 6) Подключение к сети Интернет

##### **3.6.1.2. Состав программных средств**

- 1) Dart версии 2.15.1 или выше

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

**3.6.2. Обоснование выбора технических и программных средств**

Для разработки продукта использовался Flutter Framework так как он позволяет писать кроссплатформенные приложения, для которых на обеих платформах будет одинаковый пользовательский интерфейс. Так как все приложения написанные с Flutter Framework компилируются в машинный код, который использует механизмы визуализации, встроенные в C и C++, приложения получаются очень быстрыми и высокопроизводительными.

Язык программирования Dart был выбран, потому что изначально базировался как инструмент для создания клиентских приложений. Поэтому он оптимизирован под разработку пользовательского интерфейса.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

#### 4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

##### 4.1. Ориентировочная экономическая ценность

Расчет ориентировочной экономической ценности клиентской части мобильного приложения «Stoady» не предусмотрен. Расчет экономической ценности мобильного приложения «Stoady» в целом приведен в настоящем Техническом задании «Мобильное приложение для помощи в запоминании теоретической части дисциплин».

##### 4.2. Предполагаемая потребность

Данное приложение будет использоваться как студентами для подготовки к различного рода теоретическим тестам, коллоквиумам и прочим элементам контроля, так и другими, ответственными студентами или преподавателями для подготовки этих самых тестов, определений и прочего.

Приложение централизует информацию, необходимую для подготовки к экзамену, позволит студентам получать только ту информацию, которая изучается в их вузе (посредством команд, при условии, что в вузе или группе есть ответственные люди, которые будут заполнять нужные темы и добавлять теорию). Это упростит подготовку к экзаменам, ускорит процесс обучения и позволит каждому отслеживать пробелы в знаниях.

##### 4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

Помимо перечисленных в настоящем Техническом задании «Мобильное приложение для помощи в запоминании теоретической части дисциплин» преимуществ, также можно выделить то, что приложение нацелено на вузы и другие учебные заведения для более централизованного хранения теоретических материалов и помощи студентам в усвоении теории изучаемых ими дисциплин.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



**СПИСОК ИСТОЧНИКОВ**

- 1) ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 2) ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 3) ГОСТ 19.103-77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 4) ГОСТ 19.104-78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 5) ГОСТ 19.105-78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 8) ГОСТ 19.603-78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.10-01 ТЗ 01-1				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## ПРИЛОЖЕНИЕ 1

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

Таблица 1. Описание и функциональное назначение классов.

КЛАСС	НАЗНАЧЕНИЕ
<b>AuthInfo</b>	Отвечает за информацию для регистрации нового пользователя
<b>Group</b>	Отвечает за сущность группы/команды
<b>GroupMembers</b>	Отвечает за информацию обо всех пользователях в группе/команде
<b>Logic</b>	Отвечает за хранение текущих значений всех сущностей
<b>Member</b>	Отвечает за сущность пользователя, как участника группы/команды
<b>Question</b>	Отвечает за сущность вопроса
<b>Avatar</b>	Отвечает за аватар пользователя
<b>Result</b>	Отвечает сущность результатов тестирования
<b>Statistics</b>	Отвечает за хранение всех результатов тестирования
<b>Subject</b>	Отвечает за сущность предмета
<b>TopicInfo</b>	Отвечает за сущность темы
<b>TopicTest</b>	Отвечает за список всех вопросов, относящихся к темам
<b>User</b>	Отвечает за сущность пользователя
<b>UserGroup</b>	Отвечает за информацию о пользователе в группе/команде
<b>RoundedInputTextField</b>	Отвечает за текстовое поле с иконкой-подсказкой слева

<b>TextFieldContainer</b>	Отвечает за контейнер для текстового поля, чтобы его рамка была с круглыми границами
<b>AnswerTextFieldContainer</b>	Отвечает за текстовое поле, которое перекрашивается в зависимости от введенного текста
<b>RoundedPasswordField</b>	Отвечает за текстовое поле, которое автоматически скрывает текст
<b>NothingFoundWidget</b>	Отвечает экран ошибки, если пользователей пытается просмотреть предметы не выбрав тему
<b>MenuListTile</b>	Отвечает за сущность элемента меню
<b>RoundedButton</b>	Отвечает за кнопку с округленными концами и параметрами темы
<b>SideMenu</b>	Отвечает за создание бокового меню
<b>SavedStar</b>	Отвечает за виджет звезды для добавления/удаления из сохраненных
<b>NoInternetWidget</b>	Отвечает за экран ошибки, если у пользователя нет интернета
<b>AdminModePage: Background</b>	Отвечает за интерфейс заднего плана экрана режима администратора
<b>AdminModePage: Body</b>	Отвечает за интерфейс передний плана экрана режима администратора
<b>AdminModePage: PersonCard</b>	Сущность для отображения элементов списка в режиме администратора
<b>AdminModePage: AdminModePage</b>	Отвечает за создание экрана режима администратора
<b>Admin_mode: Filter</b>	Отвечает за фильтрацию студентов на экране режима администратора
<b>QuestionManager: QuestionManagerPage</b>	Отвечает за создания экрана для управления вопросами
<b>QuestionManager: Background</b>	Отвечает за элементы интерфейса на экране управления вопросами
<b>QuestionCard</b>	Сущность для отображения списка вопросов на экране управления темами
<b>TopicCard</b>	Сущность для отображения списка тем на экране управления предметами

<b>SubjectManager: Background</b>	Отвечает за интерфейс заднего плана экрана управления предметами
<b>SubjectManager: Body</b>	Отвечает за интерфейс передний плана экрана управления предметами
<b>TopicManager: Background</b>	Отвечает за интерфейс заднего плана экрана управления темами
<b>TopicManager: Body</b>	Отвечает за интерфейс передний плана экрана управления темами
<b>Avatar: AvatarRow</b>	Отвечает за создание ряда изображений
<b>Avatar: Body</b>	Отвечает за интерфейс передний плана экрана выбора аватара
<b>Avatar: AvatarPage</b>	Отвечает за создание экрана с выбором нового аватара
<b>Group: Background</b>	Отвечает за интерфейс заднего плана экрана групп пользователя
<b>Group: Body</b>	Отвечает за интерфейс передний плана экрана групп пользователя
<b>Group: GroupCard</b>	Сущность для отображения списка групп на экране с группами пользователя
<b>Subjects: Background</b>	Отвечает за интерфейс заднего плана экрана предметов в группе
<b>Subjects: Body</b>	Отвечает за интерфейс передний плана экрана предметов в группе
<b>Subjects: SubjectCard</b>	Сущность для отображения списка предметов на экране предметов группы
<b>LearningCard</b>	Отвечает за виджет карточки с вопросом
<b>LearningPage</b>	Отвечает за создание экрана изучения
<b>ScoreCard</b>	Сущность для отображения результатов тестирования
<b>StatisticsPage</b>	Отвечает за создание экрана для просмотра статистики
<b>SubjectPage</b>	Отвечает за создание экрана создания предмета

<b>TestingPage: Background</b>	Отвечает за интерфейс экрана тестирования
<b>TestingPage</b>	Отвечает за создание экрана тестирования
<b>TopicPage</b>	Отвечает за создание экрана тем

## ПРИЛОЖЕНИЕ 2

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ

Таблица 2. Описание полей класса *AuthInfo*.

ИМЯ	ТИП	НАЗНАЧЕНИЕ	
ПОЛЯ			
email	String	Почта пользователя	
password	String	Пароль пользователя	
avatarId	int	Идентификатор аватара пользователя	
userName	String	Имя пользователя	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	AuthInfo	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 3. Описание полей и методов класса *Group*.

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
name	String	Название группы
picture	String	Аватар группы

subjects	List<Subject>	Список предметов команды	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
getName	String	-	Возвращать только первые 11 символов, если имя длиннее
fromJson	Group	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 4. Описание полей и методов класса *GroupMembers*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
members		List<UserGroup>	Хранит список всех групп пользователя
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	GroupMembers	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 5. Описание полей и методов класса *Logic*.

<b>ПОЛЯ</b>		
<b>ИМЯ</b>	<b>ТИП</b>	<b>НАЗНАЧЕНИЕ</b>
<b>currentUser</b>	<b>User</b>	Текущий пользователь
<b>userInfo</b>	<b>AuthInfo</b>	Информация, указанная пользователем при авторизации

<b>registerInfo</b>	<b>AuthInfo</b>	Информация, указанная пользователем при регистрации	
<b>currentGroupId</b>	<b>int</b>	Идентификатор выбранной пользователем группы	
<b>currentGroup</b>	<b>Group</b>	Информация о выбранной пользователем группе	
<b>currentSubjectId</b>	<b>int</b>	Идентификатор выбранного пользователем предмета	
<b>currentSubject</b>	<b>Subject</b>	Информация о выбранном пользователем предмете	
<b>currentTopic</b>	<b>TopicInfo</b>	Информация о выбранной пользователем теме	
<b>questions</b>	<b>List&lt;Question&gt;</b>	Список вопросов для изучения или тестирования	
<b>results</b>	<b>List&lt;results&gt;</b>	Список полученных результатов после тестирования	
<b>members</b>	<b>TeamMembers</b>	Список всех участников текущей группы	
<b>userGroups</b>	<b>List&lt;UserGroup&gt;</b>	Список групп пользователя	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
isAdmin	bool	-	Проверяет, что текущий пользователь является администратором в текущей группе
addIndex	void	bool add	Прибавляет/отнимает 1 от значения счетчика в пределах количества вопросов в теме



Таблица 6. Описание полей и методов класса *Member*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
id		int	Идентификатор участника группы
username		String	Имя пользователя
email		String	Почта пользователя
role		Role	Роль пользователя команды
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
isAdmin	bool	-	Проверять, является ли пользователь создателем или администратором
fromJson	Member	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 7. Описание полей и методов класса *Question*.

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
id	int	Идентификатор вопроса
questionText	String	Текст вопроса
answerText	String	Текст ответа

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	Question	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект (только с id)
fromJsonWithInfo	Question	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 8. Описание полей и методов класса Avatar.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
random	Random		Генератор случайных чисел
_frogs	List<String>		Список всех возможных аватаров
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
getRandomAvatar	int	-	Генерирует случайный номер аватара
getAvatar	String	int index	Возвращает ссылку на аватар по передаваемому индексу
getPhotos	List<String>	-	Возвращает список ссылок на возможные аватары

Таблица 9. Описание полей и методов класса *Result*.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
topicId	int		Идентификатор темы, по которой был пройден тест
name	String		Название темы, по которой был пройден тест
result	int		Результат тестирования
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	Result	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 10. Описание полей и методов класса *Statistics*.

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
results	List<Result>	Список всех результатов тестирования	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	Statistics	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 11. Описание полей и методов класса *Subject*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
id		int	Идентификатор предмета
name		String	Имя предмета
description		String	Описание предмета
topics		List<TopicInfo>	Список предметов
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	Statistics	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект (без списка предметов)
fromJsonWithSubjects	Statistics	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект
getTitle	String	-	Возвращает имя предмета
getDescription	String	-	Возвращает описание предмета

Таблица 12. Описание полей и методов класса TeamMember.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
members	List<Member>		Список всех участников команды
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	TeamMembers	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект
anyAdmin	bool	-	Проверяет, что среди участников есть хотя бы один администратор
changeAbility	void	Member member	Меняет роль участника команды на противоположную

Таблица 13. Описание полей и методов класса TopicInfo

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
id	int	Идентификатор темы
name	String	Название темы
description	String	Описание темы
questions	List<Question>	Список вопросов темы
МЕТОДЫ		

ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>fromJson</b>	<b>TopicInfo</b>	<b>dynamic json</b>	Преобразует данные из json в нужные типы и возвращает созданный объект (без вопросов)
<b>fromJsonWithQuestions</b>	<b>TopicInfo</b>	<b>dynamic json</b>	Преобразует данные из json в нужные типы и возвращает созданный объект

Таблица 14. Описание полей и методов класса *TopicTest*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
questions		List<Question>	Список вопросов
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	TopicTest	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект (Вопросы, прикреплённые к какой-то теме)
savedFromJson	TopicTest	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект (Сохраненные вопросы текущего пользователя)

Таблица 15. Описание полей и методов класса User.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
id		int	Идентификатор пользователя
name		String	Имя пользователя
email		String	Почта пользователя
password		String	Пароль пользователя
avatarId		int	Индекс аватара
saved		List<Question>	Список сохраненных вопросов
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	User	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект
registerFromJsom	User	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект
setMailPassword	void	AuthInfo info	Присваивает новые значения полям почты и пароля
getId	int	-	Возвращает id пользователя
getName	String	-	Возвращает имя пользователя

<b>getEmail</b>	<b>String</b>	-	Возвращает почту пользователя
<b>getAvatar</b>	<b>String</b>	-	Возвращает аватар пользователя
<b>isSaved</b>	<b>bool</b>	<b>Question question</b>	Проверяет, что вопрос находится в сохраненных

Таблица 16. Описание полей и методов класса *UserGroup*.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
teamId	int		Идентификатор команды
role	Role		Роль пользователя в команде
teamName	String		Имя команды
teamAvatar	String		Аватар команды
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
fromJson	TeamMembers	dynamic json	Преобразует данные из json в нужные типы и возвращает созданный объект
iAdmin	bool	-	Проверяет, что пользователь является администратором команды



Таблица 17. Описание полей и методов класса *RoundedInputTextField*

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
isBig	bool	Должно ли текстовое поле быть больше стандартного размера	
isQuestion	bool	Является ли текстовое поле вопросом	
hintText	String	Текст-подсказка внутри текстового поле	
initialValue	String	Изначальное значение внутри текстового поля	
onChanged	ValueChanged<String>	Функция, которая будет выполнена при изменении значения внутри текстового поля	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает текстовое поле в зависимости от введенных значений.

Таблица 18. Описание полей и методов класса *TextFieldManagerContainer*

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
<b>child</b>	<b>Widget</b>	Элемент, который будет находиться внутри контейнера
<b>isBig</b>	<b>bool</b>	Должен ли контейнер быть больше стандартного размера
<b>isQuestion</b>	<b>bool</b>	Является ли текстовое поле вопросом

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает контейнер для текстового поля в зависимости от введенных значений.

Таблица 19. Описание полей и методов класса AnswerTextFieldContainer.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
currentState		AnswerState	Текущий статус поля, в зависимости от которого поле будет иметь разные цвета
child		Widget	Элемент, который будет находиться внутри контейнера
isSamll		bool	Должен ли контейнер быть меньше стандартного размера
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает контейнер для текстового поля в зависимости от введенных значений.

Таблица 20. Описание полей и методов класса RoundedPasswordField.

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
onChaged	ValueChanged<String>	Функция, которая будет выполнена при каждом изменении значения внутри текстового поля

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает контейнер для текстового поля для пароля, в котором автоматически будут скрываться символы, в зависимости от введенных значений.

Таблица 21. Описание полей и методов класса *NothingFoundWidget*.

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
text	String	Текст ошибки, который будет выводиться на экране	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран ошибки, на котором будет написана причина возникновения – что-то не имеет значения.

Таблица 22. Описание полей и методов класса *MenuListTile*.

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
<b>press</b>	<b>VoidCallback</b>	Функция, которая будет выполнена при нажатии на раздел меню.

<b>text</b>	<b>String</b>	Текст, который отображается, как раздел меню	
<b>icon</b>	<b>IconData</b>	Иконка, соответствующая разделу меню	
<b>canClick</b>	<b>bool</b>	Может ли пользователь нажать на данный элемент меню	
МЕТОДЫ			
<b>ИМЯ</b>	<b>ТИП ВОЗВРАЩАЕМОГО</b>	<b>АРГУМЕНТЫ</b>	<b>НАЗНАЧЕНИЕ</b>
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает раздел меню

Таблица 23. Описание полей и методов класса *RoundedButton*.

<b>ПОЛЯ</b>		
<b>ИМЯ</b>	<b>ТИП</b>	<b>НАЗНАЧЕНИЕ</b>
<b>text</b>	<b>String</b>	Текст кнопки
<b>press</b>	<b>VoidCallBack</b>	Функция, которая будет выполняться при нажатии на кнопку
<b>color</b>	<b>Color</b>	Основной цвет кнопки
<b>textColor</b>	<b>Color</b>	Цвет границ кнопки
<b>reverse</b>	<b>bool</b>	Надо ли менять основной цвет кнопки с цветом границ
<b>borders</b>	<b>bool</b>	Нужны ли границы в кнопке
<b>isSmall</b>	<b>bool</b>	Является ли кнопка меньше стандартного размера
<b>МЕТОДЫ</b>		

ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает кнопку в соответствии с заданными значениями
<b>newElevatedButton</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает кнопку стандартного размера
<b>smallElevatedButton</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает кнопку меньше стандартного размера

Таблица 24. Описание полей и методов класса *SideMenu*.

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает боковое меню

Таблица 25. Описание полей и методов класса *SavedStar*.

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает виджет пятиконечной звезды в зависимости от того добавлен ли вопрос в сохраненные
<b>addQuestion</b>	<b>Future&lt;void&gt;</b>	<b>int questionId, BuildContext context</b>	Отправляет серверу запрос на добавление вопроса в сохраненное
<b>removeQuestion</b>	<b>Future&lt;void&gt;</b>	<b>int questionId, BuildContext context</b>	Отправляет серверу запрос на удаление вопроса в сохраненное

Таблица 26. Описание полей и методов класса NoInternetWidget.

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран с ошибкой, если у пользователя нет интернета

Таблица 27. Описание полей и методов admin\_mode: Background.

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
child	Widget	Виджет состоящий из всего, что было прописано в классе body экрана	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Отрисовывает основные элементы заднего плана экрана
buildItem	Widget	BuildContext context, Int index	Отображает список участников в зависимости от фильтрации
changeNameAndAvatar	Future<void>	BuildContext context, String name, String avatar	Отправляет на сервер запрос на изменение имени и аватара группы

Таблица 28. Описание полей и методов admin\_mode: Body

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ

<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Метод для отрисовки основных элементов экрана режима администратора
<b>drawFilterChip</b>	<b>Widget</b>	<b>String query, BuildContext context</b>	Создает кнопки для фильтрации участников

Таблица 29. Описание полей и методов *admin\_mode: PersonCard*.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
user	Member		Пользователь, для которого создается карточка
isCreator	bool		Является ли пользователь создателем команды
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает элемент списка пользователей для экрана администратора
deleteBackgroundItem	Widget	-	Отображает задний фон при удалении пользователя
removeUser	Future<void>	BuildContext context, int userId	Отправляет на сервер запрос на удаление пользователя
changeUserRole	Future<void>	Member user, BuildContext context	Отправляет на сервер запрос на изменение роли пользователя

Таблица 30. Описание полей и методов *admin\_mode: AdminModePage*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
_fitireList		Future<List<Member>>	Хранит список участников группы
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран администратора
getMembers	Future<List<Member>>	-	Отправляет на сервер запрос для получения списка участников группы

Таблица 31. Описание полей и методов *admin\_mode: Filter*.

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
_showAdmin		bool	Нужно ли отображать в списке участников администраторов
_showStudents		bool	Нужно ли отображать в списке участников студентов
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
getBool	bool	String query	Проверяет, что отправленный запрос был на администратора
changeBool	void	String query	Проверяет, что отправленный запрос был на студента



<b>getString</b>	<b>String</b>	<b>String query</b>	Возвращает текст кнопки в зависимости от запроса
<b>getAdmin</b>	<b>bool</b>	-	Возвращает значение переменное <b>_showAdmin</b>
<b>getStudent</b>	<b>bool</b>	-	Возвращает значение переменное <b>_showStudent</b>

Таблица 32. Описание полей и методов *QuestionManagerPage*

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
isNew	bool		Показывает, является ли данный вопрос новым
_currentTopics	Future<List<TopicInfo>>		Хранит список тем
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран управления вопросами
getTopics	Future<List<TopicInfo>>	-	Отправляет на сервер запрос на получение списка тем

Таблица 33. Описание полей и методов *question\_manager: Background*

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
<b>isNew</b>	<b>bool</b>	Показывает, является ли вопрос новым

<b>_questionText</b>	<b>String</b>	Хранит текст вопроса	
<b>_answerText</b>	<b>String</b>	Хранит текст ответа	
<b>МЕТОДЫ</b>			
<b>ИМЯ</b>	<b>ТИП ВОЗВРАЩАЕМОГО</b>	<b>АРГУМЕНТЫ</b>	<b>НАЗНАЧЕНИЕ</b>
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает интерфейс для экрана управления вопросами
<b>buttons</b>	<b>Widget</b>	<b>Size size, BuildContext context, bool isNew</b>	Создает кнопки сохранения и отмены изменений
<b>changeQuestion</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос для изменения вопроса
<b>addQuestion</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос на добавление вопроса
<b>deleteQuestion</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос на удаление вопроса
<b>confirmDelete</b>	<b>AlertDialog</b>	<b>BuildContext context</b>	Отображает диалоговое окно с подтверждением удаления вопроса

Таблица 34. Описание полей и методов класс *QuestionCard*

<b>ПОЛЯ</b>		
<b>ИМЯ</b>	<b>ТИП</b>	<b>НАЗНАЧЕНИЕ</b>
<b>question</b>	<b>Question</b>	Хранит текущий вопрос
<b>press</b>	<b>VoidCallback</b>	Функция, которая будет вызвана при нажатии на вопрос
<b>МЕТОДЫ</b>		

ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает карту с вопросом

Таблица 35. Описание полей и методов TopicCard.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
topic	TopicInfo		Хранит текущую тему
press	VoidCallback		Функция, которая будет вызвана при нажатии на тему
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает карту с темой

Таблица 36. Описание полей и методов SubjectManager: Background

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
<b>child</b>	<b>Widget</b>	Виджет состоящий из всего, что было прописано в классе body экрана
<b>isNew</b>	<b>bool</b>	Показывает, является ли предмет новым
<b>_name</b>	<b>String</b>	Название предмета
<b>_description</b>	<b>String</b>	Описание предмета

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает интерфейс для экрана управления предметами
<b>buttons</b>	<b>Widget</b>	<b>Size size, BuildContext context, bool isNew</b>	Создает кнопки сохранения и отмены изменений
<b>changeSubject</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос для изменения предмета
<b>addSubject</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос на добавление предмета
<b>deleteSubject</b>	<b>Future&lt;void&gt;</b>	<b>BuildContext context</b>	Отправляет на сервер запрос на удаление предмета
<b>confirmDelete</b>	<b>AlertDialog</b>	<b>BuildContext context</b>	Отображает диалоговое окно с подтверждением удаления предмета

Таблица 37. Описание полей и методов *SubjectManager*: *Body*

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
isNew		bool	Показывает, является ли предмет новым
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает основной интерфейс экрана управления предметами

Таблица 38. Описание полей и методов TopicManager: Background.

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
child	Widget	Виджет состоящий из всего, что было прописано в классе body экрана	
isNew	bool	Показывает, является ли тема новой	
_name	String	Название темы	
_description	String	Описание темы	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает интерфейс для экрана управления темами
buttons	Widget	Size size, BuildContext context, bool isNew	Создает кнопки сохранения и отмены изменений
changeTopic	Future<void>	BuildContext context	Отправляет на сервер запрос для изменения темы
addTopic	Future<void>	BuildContext context	Отправляет на сервер запрос на добавление темы
deleteTopic	Future<void>	BuildContext context	Отправляет на сервер запрос на удаление темы
confirmDelete	AlertDialog	BuildContext context	Отображает диалоговое окно с подтверждением удаления темы

Таблица 39. Описание полей и методов TopicManager: Body

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
isNew	bool		Показывает, является ли тема новой
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает основной интерфейс экрана управления темами

Таблица 40. Описание полей и методов Avatar: AvatarRow.

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
photos	List<String>		Хранит ссылки на изображения
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает список аватаров на выбор
createAvatar	Widget	BuildContext context, int index	Создает элемент списка изображений
buildRow	Widget	BuildContext context, Size size, int index	Создает строку изображений
changeAvatar	Future<void>	int newIndex	Отправляет на сервер запрос на смену аватара пользователя

Таблица 41. Описание полей и методов Avatar: Body

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает основной интерфейс экрана выбора аватара

Таблица 42. Описание полей и методов Avatar: AvatarPage

МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает страницу выбора аватара

Таблица 43. Описание полей и методов Group: Background

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
child		Widget	Виджет состоящий из всего, что было прописано в классе body экрана
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает список групп

Таблица 44. Описание полей и методов Group: Body

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
maxNameLen	int	Максимальная возможная длина на имени новой команды	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает необходимые для создания группы элементы интерфейса

Таблица 45. Описание полей и методов Group: GroupCard

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
avatar	String		Изображение, соответствующее группе
name	String		Имя группы
press	VoidCallBack		Функция, которая выполнится при нажатии на элемент списка
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает карточку группы для списка команд пользователя



Таблица 46. Описание полей и методов Subjects: Background

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
child	Widget		Виджет состоящий из всего, что было прописано в классе body экрана
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает задний план для экрана выбора предмета

Таблица 47. Описание полей и методов Subjects: Body

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
showAddButton		bool	Нужно ли показывать кнопку добавления темы
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает список со всеми доступными в выбранной команде предметами

Таблица 48. Описание полей и методов Subjects: SubjectCard

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
subject	Subject		Текущий предмет
press	VoiceCallback		Функция, которая будет выполнена при нажатии на элемент списка
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает карточку предмета для общего списка всех доступных в выбранной команде предметов

Таблица 49. Описание полей и методов LearningCard

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
showFrontSize	bool		Показывает, какая сторона карточки сейчас должна быть показана на экране
isTestingCard	bool		Является ли карточки тестовой
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает контейнер для карточки
buildFlipAnimation	Widget	-	Создает анимацию переворачивающиеся карточки

<b>__transitionBuilder</b>	<b>Widget</b>	<b>Widget widget, Animation&lt;double&gt; animation</b>	Поворачивает карточку относительно оси Y
<b>_buildFront</b>	<b>Widget</b>	-	Отображает переднюю часть карточки - вопрос
<b>_buildRear</b>	<b>Widget</b>	-	Отображает заднюю часть карточки - ответ
<b>_buildLayout</b>	<b>Widget</b>	<b>Key key, String faceName, Color backgroundColor</b>	Создает основу для карточки

Таблица 50. Описание полей и методов LearningPage

ПОЛЯ			
ИМЯ		ТИП	НАЗНАЧЕНИЕ
isSaved		bool	Проходил ли изучение по сохраненным вопросам
_futureQuestions		Future<List<Question>>	Список вопросов, по которым будет запущен экран изучения
_futureSavedQuestions		Future<List<Question>>	Список сохраненных вопросов пользователя
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран изучения вопросов
getQuestions	Future<List<Question>>	-	Отправляет на сервер запрос для получения вопросов по текущей теме
getSavedQuestions	Future<List<Question>>	-	Отправляет на сервер запрос на получение всех сохраненных вопросов текущего пользователя

Таблица 51. Описание полей и методов ScoreCard

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
currentResult	Result		Текущий результат тестирования в процентах
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает элемент списка для статистики

Таблица 52. Описание полей и методов StatisticsPage

ПОЛЯ			
ИМЯ	ТИП	НАЗНАЧЕНИЕ	
<code>_futureResults</code>	<code>Future&lt;List&lt;Result&gt;&gt;</code>	Список всех результатов пользователя	
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<code>build</code>	<code>Widget</code>	<code>BuildContext context</code>	Создает экран просмотра статистики
<code>getResult</code>	<code>Future&lt;List&lt;Result&gt;&gt;</code>	-	Отправляет на сервер запрос на получения всех результатов тестирования текущего пользователя

Таблица 53. Описание полей и методов SubjectPage

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
_currentSubject	Future<Subject>		Текущий предмет
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран с подробной информацией о предмете
getSubject	Future<Subject>	-	Отправляет на сервер запрос для получения информации о предмете

Таблица 54. Описание полей и методов TestingPage: Background

ПОЛЯ		
ИМЯ	ТИП	НАЗНАЧЕНИЕ
<code>child</code>	<code>Widget</code>	Виджет состоящий из всего, что было прописано в классе <code>body</code> экрана
<code>isSaved</code>	<code>bool</code>	Показывает, проходит ли сейчас тестирование по сохранённым вопросам
<code>currentAnswer</code>	<code>String</code>	Текущий ответ, который пользователь ввел
<code>points</code>	<code>Map&lt;Question, int&gt;</code>	Словарь результатов тестирования по каждому вопросу
<code>totalScore</code>	<code>int</code>	Финальный результат тестирования по текущей теме
МЕТОДЫ		

ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<b>build</b>	<b>Widget</b>	<b>BuildContext context</b>	Создает основные элементы интерфейса для экрана тестирования
<b>calcResult</b>	<b>int</b>	-	Считает результаты тестирования
<b>saveResults</b>	<b>Future&lt;void&gt;</b>	<b>int newResult</b>	Отправляет на сервер запрос с сохранением итоговых результатов

Таблица 55. Описание полей и методов *TestingPage*

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
isSaved	bool		Проходит ли сейчас тестирование по сохранённым вопросам
_futureQuestions	Future<List<Question>>		Список вопросов, по которым будет запущен экран тестирования
_futureSavedQuestions	Future<List<Question>>		Список сохраненных вопросов пользователя
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
build	Widget	BuildContext context	Создает экран тестирования по текущей теме
getQuestions	Future<List<Question>>	-	Отправляет на сервер запрос для получения вопросов по текущей теме
getSavedQuestions	Future<List<Question>>	-	Отправляет на сервер запрос на получение всех сохраненных

			вопросов текущего пользователя
--	--	--	--------------------------------

Таблица 56. Описание полей и методов TopicPage

ПОЛЯ			
ИМЯ	ТИП		НАЗНАЧЕНИЕ
<code>_currentTopic</code>	<code>Future&lt;TopicInfo&gt;</code>		Текущая тема
МЕТОДЫ			
ИМЯ	ТИП ВОЗВРАЩАЕМОГО	АРГУМЕНТЫ	НАЗНАЧЕНИЕ
<code>build</code>	<code>Widget</code>	<code>BuildContext context</code>	Создает экран с информацией о текущей теме
<code>getTopic</code>	<code>Future&lt;TopicInfo&gt;</code>	-	Отправляет на сервер запрос для получения информации о текущей теме

## ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

[illegible]