**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка для Android-приложения для мозгового штурма.

В разделе «Введение» указаны полное и краткое наименование программы, а также документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указывают назначение программы, а также краткую характеристику области применения программы.

Раздел «Технические характеристики» состоит из следующих разделов: постановка задачи на разработку программы, описание алгоритма и функционирования программы, обоснование выбора метода организации входных и выходных данных, обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указаны предполагаемая потребность и экономические преимущества разработки по сравнению с аналогами.

В разделе «Источники, использованные при разработке» указан перечень материалов и документов, которые использовались при разработке.

Настоящий документ разработан в соответствии с требованиями:

ГОСТ 19.101-77 Виды программ и программных документов [[1]](#ГОСТ_19_101), ГОСТ 19.103-77 Обозначения программ и программных документов [[3]](#ГОСТ_19_103), ГОСТ 19.104-78 Основные надписи [[4]](#ГОСТ_19_104), ГОСТ 19.105-78 Общие требования к программным документам [[5]](#ГОСТ_19_105), ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [[6]](#ГОСТ_19_106), ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению [[12]](#ГОСТ_19_404).

**СОДЕРЖАНИЕ**

[1. ВВЕДЕНИЕ 4](#_Toc195103564)

[1.1. Наименование программы 4](#_Toc195103565)

[1.2. Документ, на основании которого ведется разработка 4](#_Toc195103566)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 5](#_Toc195103567)

[2.1. Функциональное назначение программы 5](#_Toc195103568)

[2.2. Эксплуатационное назначение 5](#_Toc195103569)

[2.3. Краткая характеристика области применения 5](#_Toc195103570)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 6](#_Toc195103571)

[3.1. Постановка задачи на разработку программы 6](#_Toc195103572)

[3.2. Описание и обоснование архитектуры программы 7](#_Toc195103573)

[3.2.1. Архитектура программы и использованные библиотеки 7](#_Toc195103574)

[3.2.2. Архитектура базы данных 9](#_Toc195103575)

[3.3. Описание алгоритма и функционирования программы 12](#_Toc195103576)

[3.3.1. Регистрация 14](#_Toc195103577)

[3.3.2. Авторизация 14](#_Toc195103578)

[3.3.3. Смена пароля 14](#_Toc195103579)

[3.3.4. Выход из аккаунта 14](#_Toc195103580)

[3.3.5. Просмотр завершенных мозговых штурмов 15](#_Toc195103581)

[3.3.6. Экспорт данных о завершенном мозговом штурме в PDF 15](#_Toc195103582)

[3.3.7. Удаление мозгового штурма 15](#_Toc195103583)

[3.3.8. Создание виртуальной комнаты для мозгового штурма 16](#_Toc195103584)

[3.3.9. Присоединение к виртуальной комнате для мозгового штурма 16](#_Toc195103585)

[3.3.10. Функции, работающие с помощью WebSocket 16](#_Toc195103586)

[3.3.11. Просмотр идей 17](#_Toc195103587)

[3.3.12. Выход из комнаты 17](#_Toc195103588)

[3.3.13. Удаление комнаты 17](#_Toc195103589)

[3.3.14. Завершение мозгового штурма 17](#_Toc195103590)

[3.3.15. Обмен идей в формате Round Robin 18](#_Toc195103591)

[3.4. Описание и обоснование выбора метода организации входных и выходных данных 18](#_Toc195103592)

[3.5. Описание и обоснование выбора состава технических и программных средств 19](#_Toc195103593)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 21](#_Toc195103594)

[4.1. Ориентировочная экономическая эффективность 21](#_Toc195103595)

[4.2. Предполагаемая потребность 21](#_Toc195103596)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 24](#_Toc195103597)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 26](#_Toc195103598)

# ВВЕДЕНИЕ

## Наименование программы

**Наименование темы разработк**и: Android-приложение для мозгового штурма.

**Краткое наименование программы:** «StormMaster».

## Документ, на основании которого ведется разработка

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия», а также утвержденная академическим руководителем тема курсового проекта.

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## Функциональное назначение программы

Функциональным назначением приложения является возможность создания виртуальных комнат с чатом для проведения различных вариаций мозгового штурма, возможность предлагать свои идеи, возможность другими пользователями голосовать за понравившиеся идеи, возможность формирования отчета на основе этих данных, а также хранение этих отчетов внутри приложения для офлайн доступа к ним.

## Эксплуатационное назначение

Программа должна помогать пользователю организовать мозговые штурмы, позволять пользователям обмениваться идеями в реальном времени, а также хранить и формировать их результаты в виде отчёта. Приложение предназначено для широкой аудитории. Предположительно конечными пользователями являются люди, работающие над каким-то проектом или задачей в команде.

## Краткая характеристика области применения

В современном мире у человека любой сферы деятельности может возникнуть необходимость придумать какую-либо идею для реализации собственных планов или планов своей команды. Это может быть тема для курсовой работы, идея нового приложения, которого ещё нет на рынке, или, например, способ решения какой-либо проблемы. Однако, этот процесс придумывания идей может оказаться достаточно сложным. Для его упрощения был придуманы и предложены различные методики, одна из которых – мозговой штурм.

Мозговой штурм – это методика, которая позволяет коллективно генерировать идеи и искать решения определённой проблемы. Мозговой штурм имеет несколько разновидностей.

Мобильное приложение «StormMaster» призвано помочь организовать его различные вариации, а также хранить и упорядочивать идеи, которые были предложены во время его проведения.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Android-приложение для мозгового штурма должно предоставлять пользователям следующий функционал:

1. Регистрация с использованием никнейма и пароля;
2. Авторизация;
3. Смена пароля;
4. Выход из аккаунта, как локально, так и со всех устройств одновременно;
5. Просмотр завершенных мозговых штурмов, в которых пользователь был как участник или как создатель;
6. Экспортирование завершенных мозговых штурмов в формат PDF;
7. Удаление мозгового штурма у всех пользователей, если пользователь является его создателем;
8. Создание виртуальной комнаты для проведения мозгового штурма в разных режимах мозгового штурма;
9. Присоединение к комнате, упомянутой в пункте 8;
10. Обмен сообщениями в реальном времени;
11. Отметка сообщения как идеи для их последующей группировки;
12. Голосование за идеи;
13. Просмотр сообщений, помеченных как одна идея;
14. Выход из комнаты, как участник;
15. Удаление комнаты, как её создатель;
16. Завершение мозгового штурма и формирования отчёта о проведении мозгового штурма, являясь создателем комнаты;
17. Обмен идей в формате Round Robin;

Для реализации данного функционала помимо мобильного приложения (клиента) должна быть разработана серверная часть, которая будет организовывать хранение информации пользователей, а также обмен информацией между ними.

## Описание и обоснование архитектуры программы

### Архитектура программы и использованные библиотеки

Для работы программы была выбрана клиент-серверная архитектура, позволяющая разделить обязанности между клиентом и сервером. Сервер организует аутентификацию пользователей, хранение их данных, а также позволяет пользователям обмениваться информацией в реальном времени. Клиент (мобильное приложение), в свою очередь, предоставляет удобный интерфейс для взаимодействия с сервером, позволяет локально хранить данные пользователя, позволяя пользователю работать с ними без интернет-соединения, а также предоставляет пользователю возможность экспортировать данные отчета в формат PDF.

Мобильное приложение разработано на языке Java с использованием среды Android Studio для мобильных устройств на базе Android. Android Studio имеет простую интеграцию с Android SDK, поддерживает много популярных библиотек и рекомендуется для Android-разработки компанией Google. Java – язык программирования, имеющий много общедоступных материалов и библиотек, которые можно использовать для эффективной реализации проекта.

Для разработки мобильного приложения были использованы следующие библиотеки:

1. Retrofit 2 + GSON Converter [[14]](#Retrofit)

Retrofit – библиотека для работы с HTTP-запросами, позволит упростить код для их написания. Рекомендована компанией Google для API запросов.

GSON Converter - преобразует JSON ответы сервера в Java объекты, тем самым позволяя быстро и просто обрабатывать ответы сервера.

1. Room database [[15]](#Room)

Room – ORM-библиотека для работы с SQLite в Android, представляет собой абстракцию на SQL, а также поддерживает работу с LiveData, что позволяет удобно связать базу данных и список элементов RecyclerView.

1. iText 7 [[16]](#iText)

iText – библиотека, необходимая для создания PDF документов. Позволяет использовать HTML-синтаксис для форматирования документов, что упрощает их создание с помощью кода.

1. OkHttp [[17]](#OkHttp)

OkHttp – библиотека, необходимая для работы с WebSocket соединением.

1. Material Design Components [[18]](#Material_Design_Components)

Material Design Components обеспечивает современный UI и включает в себя готовые компоненты интерфейса для использования.

Серверная часть приложения разработана на Python с использованием Django фреймворка [[19]](#Django). Django – это высокоуровневый фреймворк, который позволяет быстро разрабатывать веб-приложения (в данном случае – серверную часть) с авторизацией пользователей.

Для реализации серверной части приложения также были использованы:

1. Django REST framework (DRF) [[20]](#Django_REST)

DRF упрощает создание API (позволяет реализовывать сериализаторы данных и наборы энд поинтов), а также поддерживает работу с токенами для авторизации пользователей, что удобно для взаимодействия с мобильным приложениям.

1. Djoser [[21]](#Djoser)

Djoser предоставляет готовую реализацию энд поинтов для аутентификации. Эти энд поинты все равно требовали модификации под реализуемый проект, однако позволили значительно ускорить разработку его части, связанной с авторизацией, регистрацией, сменой пароля и выхода из аккаунта.

1. DRF-YASG [[22]](#DRF_YASG)

DRF-YASG представляет собой инструмент для автоматической генерации документации API в удобном формате. Таким образом, упрощается процесс написания документации к серверной части приложения.

1. Django Channels [[23]](#Django_Channels)

Django channels позволяет работать с WebSocket соединением и обеспечивает возможность обмениваться информацией клиенту и серверу в реальном времени.

1. Daphne [[24]](#Daphne)

Альтернативный стандартному Django сервер, который поддерживает асинхронные запросы. Рекомендован для работы с Django Channels.

Для корректной работы приложения необходима база данных, которая описана в пункте 3.2.2 «Архитектура базы данных».

### Архитектура базы данных

рис. 1

Сервер использует PostgreSQL для хранения данных пользователей.

Схема базы данных представлена на рис. 1.

Ниже описана структура базы данных, созданной для проекта, за исключением таблиц, авто генерируемых фреймворком Django.

Каждая таблица имеет поле ID – уникальный идентификатор записи в таблице.

1. Таблица *customUser\_user*

Назначение: Хранение пользователей системы. Была создана отличная от стандартной Django модель пользователя для использования собственных средств валидации пароля и никнейма.

Поля:

* username — уникальное имя пользователя (никнейм).
* password — хэш пароля.

В данной таблице есть и другие поля, но они напрямую не используются приложением и были добавлены Django фреймворком.

2. Таблица *brainstormsApp\_brainstorm*

Назначение: Хранение информации о завершенных мозговых штурмах.

Поля:

* *title* — название сессии
* *completionDate* — дата завершения сессии.
* *details* — отчёт о мозговом штурме, написанный с помощью HTML синтаксиса.
* *creator\_id* — ID создателя (внешний ключ к *customUser\_user*).

3. Таблица *brainstormsApp\_brainstorm\_participants*

Назначение: Хранение участников завершенных мозговых штурмов.

Поля:

* *brainstorm\_id* — ID сессии (внешний ключ к *brainstormsApp\_brainstorm*).
* *user\_id* — ID участника (внешний ключ к *customUser\_user*).

4. Таблица *roomApp\_room*

Назначение: Хранение информации о комнате для проведения мозгового штурма.

Поля:

* *title* — название комнаты (макс. 25 символов).
* *room\_code* — уникальный код комнаты (6 символов).
* *details* — описание комнаты (текст).
* *isChatStarted* — флаг начала чата, нужен для правильной синхронизации WebSocket соединения.
* *creator\_id* — ID создателя (внешний ключ к *customUser\_user*).
* *room\_type\_id* — ID типа комнаты (внешний ключ к *roomApp\_roomtype*).

5. Таблица *roomApp\_room\_participants*

Назначение: Хранение участников комнат.

Поля:

* *room\_id* — ID комнаты (внешний ключ к *roomApp\_room*).
* *user\_id* — ID участника (внешний ключ к *customUser\_user*).

6. Таблица *roomApp\_idea*

Назначение: Хранение информации о идеях, предложенных в комнатах.

Поля:

* + - *idea\_number* — порядковый номер идеи, которые используют пользователи для их идентификации во время проведения мозгового штурма.
    - *votes* — количество голосов за идею.
    - *room\_id* — ID комнаты (внешний ключ к *roomApp\_room*).

7. Таблица *roomApp\_idea\_voters*

Назначение: Хранение информации о том, какие пользователи проголосовали за какую идею.

Поля:

* *idea\_id* — ID идеи (внешний ключ к *roomApp\_idea*).
* *user\_id* — ID пользователя (внешний ключ к *customUser\_user*).

8. Таблица *roomApp\_message*

Назначение: Хранение сообщений в чате комнаты.

Поля:

* *text* — текст сообщения.
* *timestamp* — дата и время отправки.
* *idea\_id* — ID связанной идеи (опционально, внешний ключ к *roomApp\_idea*).
* *room\_id* — ID комнаты (внешний ключ к *roomApp\_room*).
* *sender\_id* — ID отправителя (внешний ключ к *customUser\_user*).

9. Таблица *roomApp\_roomtype*

Назначение: Хранение информации о возможных типах мозговых штурмов. Заполняется автоматически при выполнении migrate в Django. Состоит из типов «Классический» и «Round Robin».

Поля:

* *name* — название типа.

10. Таблица *roomApp\_roundrobindata*

Назначение: Хранение данных для правильный работы этапа обмена идей Round Robin

Поля:

* *completed\_rounds* — количество завершенных раундов.
* *room\_id* — ID комнаты (внешний ключ к *roomApp\_room*).

11. Таблица *roomApp\_roundrobindata\_users\_finished\_idea*

Назначение: Хранение пользователей, которые уже закончили работу над идеей в данном раунде Round Robin.

Поля:

* *roundrobindata\_id* — ID данных раунда (внешний ключ к roomApp\_roundrobindata).
* user\_id — ID пользователя (внешний ключ к *customUser\_user*).

Для удобства работы, а также доступа к данным без интернет-соединения, мобильное приложение создает уменьшенную копию таблицы *brainstormsApp\_brainstorm,* содержащую только информацию о мозговых штурмах, в которых участвовал пользователь.

Для этих же целей создается копия таблицы *roomApp\_message,* содержащая только сообщения, которые связаны с комнатой, в которой в данный момент находится пользователь. Это позволяет функции «Просмотр идей» работать с имеющимся данными даже при потере интернет-соединения.

## Описание алгоритма и функционирования программы

Мобильное приложение запускается в раздел «Авторизация», если пользователь не авторизован, или в раздел «История мозговых штурмов», если пользователь авторизован (подробнее о разделах можно почитать в п. «Состав выполняемых функций» Технического задания). Проверка, авторизован ли пользователь производится путем проверки наличия соответствующей записи в локальной памяти устройства Shared Preferences. В остальном, приложение не имеет четкой последовательности выполнения и порядок выполнения функций напрямую зависит от действий пользователя. Работа сервера зависит от запросов, которые приходят от пользователей, и также не имеет строгий порядок выполнения своих функций.

Перед отправкой любого запроса на сервер приложение сначала проверяет данные на корректность (соответствие данных, введенных в текстовое поле, требованиям для данного поля, указанным в Техническом задании) и только после этого отправляет необходимый запрос. Это сделано для того, чтобы избежать заведомо ложных запросов. Однако, сервер все равно проводит дополнительную проверку передаваемых ему данных и вернет ошибку, если они не соответствуют требованиям. Также сервер проверяет возможность совершение операции пользователем. Например, это позволяет совершать некоторые операции только создателю мозгового штурма.

Приложение умеет обрабатывать все ошибки, возникающие при API запросах, в том числе ошибки на стороне сервера и ошибки подключения к интернету, и уведомляет пользователя о них. Все API запросы, за исключением запросов авторизации и регистрации, требуют токена пользователя для его идентификации. В случае неправильного токена сервер возвращает ошибку 401, а приложение при получении такой ошибки при отправке любого запроса возвращает пользователя в раздел «Авторизация», стирает все данные приложения и просит пользователя авторизоваться снова. Приложение использует токен для авторизированных запросов, взятый из Shared Preferences.

Приложение при потере WebSocket соединения пытается переподключиться к серверу до тех пор, пока у него этого не получится. При установке соединения сервер отправляет пользователю данные синхронизации, которое приложение использует, чтобы восстановить пропущенную пользователем вовремя отсутствия подключения информацию и переместить его в другой раздел, при необходимости.

Описание необходимых действий пользователя для вызова определенных функций приложения и отправки запроса на сервер описаны в п. «Выполнение программы» Руководства оператора. Ниже описан лишь принцип работы функций.

### Регистрация

При попытке регистрации отправляется API запрос на сервер.

В случае успешной регистрации, сервер создаст нового пользователя и вернет приложению токен авторизации, который будет создан непосредственно после регистрации. Приложение сохранит токен и никнейм в Shared Preferences для дальнейшего использования и перейдет в раздел «Истории мозговых штурмов».

### Авторизация

При попытке авторизации отправляет API запрос на сервер.

Если пользователя с введенными данными нет, пользователь уведомляется об этом.

В случае успешной авторизации, сервер возвращает токен, который используется для идентификации данного пользователя (если токена на данный момент нет, то он создается). Приложение сохранит токен и никнейм в Shared Preferences для дальнейшего использования и перейдет в раздел «Истории мозговых штурмов».

### Смена пароля

При попытке смена пароля отправляется API запрос на сервер.

В случае успешного смена пароля, сервер меняет пароль в базе данных, а приложение уведомляет об этом пользователя и возвращает его в раздел «Профиль».

### Выход из аккаунта

Если пользователь выбирает обычный выход из аккаунта, то приложение стирает все данные пользователя и переходит к разделу «Авторизация».

Если пользователь выбирает опцию «Выйти со всех устройств», то приложение делает API запрос на сервер с целью удалить токен авторизации, таким образом сделав запросы с этим токеном не валидными и заставив приложения, использующие этот же токен (аккаунт), выйти из аккаунта. В случае успеха, помимо удаления токена с сервера, происходит тоже самое, что и при обычном выходе из аккаунта.

### Просмотр завершенных мозговых штурмов

При открытии раздела «История мозговых штурмов» первым делом в отображаемый список загружаются данные о завершенных мозговых штурмах, хранящиеся в локальной базе данных для того, чтобы пользователь мог взаимодействовать с ними, даже, если у него отсутствует интернет.

Затем на сервер отправляется API запрос с целью получения завершенных мозговых штурмов пользователя, в которых он являлся участником или создателем. В случае успеха, в базу данных (и непосредственно в список) добавляются мозговые штурмы, данных о которых нет в локальной базе приложения (уже имеющиеся данные не имеет смысла обновлять, поскольку после завершения мозгового штурма информация о нем в базе данных не меняется и остается статичной, поэтому происходит только добавление). Помимо добавления, приложение также просматривает наличие в локальном хранилище данных о мозговом штурме, которых нет в ответе сервера. Это означает, что эти данные были удалены создателем мозгового штурма и подлежат непосредственному удалению и из локальной базы данных пользователя.

### Экспорт данных о завершенном мозговом штурме в PDF

После выбора пользователем место для сохранения экспортирования файла. Приложение создает файл на основе имеющихся данных, включая титульный лист и информацию о идеях. При этом сам текст идей переформатируется в HTML формат, согласно требованиям библиотеки iText.

После успешного создания файла приложение открывает его с использованием стандартного сторонних приложений для открытия PDF файла. Стандартное приложение для открытия PDF файлов определяется пользователем устройства в его настройках.

В случае невозможности создания файла пользователь уведомляется об этом.

Данная функция работает без подключения к серверу.

### Удаление мозгового штурма

Данная функция доступна только создателю мозгового штурма.

При попытке удаления мозгового штурма отправляется API запрос. При успешном запросе, данные о мозговом штурме удаляются из базы данных, а также удаляются у всех пользователей (принцип этого описан в п. 3.3.5). После этого пользователь уведомляется об этом и переходит к разделу «История мозговых штурмов».

### Создание виртуальной комнаты для мозгового штурма

После введения названия мозгового штурма и выбора его типа выполняется API запрос для создания новой комнаты.

В случае успеха, создается новая комната в базе данных сервера с уникальным 6-ти символьным кодом, состоящим из букв латинского алфавита, а также имеет тип комнаты, выбранный пользователем. Генерация этого кода происходит случайным образом на стороне сервера до тех пор, пока сгенерированный код не будет уникальным в таблице. Сгенерированный код возвращается приложению в ответе сервера, а приложение сохраняет его и переход к разделу «Запуск мозгового штурма», где подключается с помощью имеющихся данных к серверу через WebSocket соединение для последующего обновления данных в реальном времени.

В случае, если тип комнаты «Round Robin», сервер также создает round\_robin\_data, связанный с этой комнатой, а также создает идею и сообщение для каждого участника мозгового штурма. Номер идеи присваивается как порядковый номер пользователя в списке участников мозгового штурма. Именно с этой идеи и сообщения участник мозгового штурма начнет работу в рамках этапа «Round Robin».

### Присоединение к виртуальной комнате для мозгового штурма

При попытке присоединения к мозговому штурму выполняется API запрос.

В случае успеха, приложение переходит в раздел «Запуск мозгового штурма», где подключается с помощью имеющихся данных к серверу через WebSocket соединение для последующего обновления данных в реальном времени.

### Функции, работающие с помощью WebSocket

Функции обмена сообщений в реальном времени, отметки сообщения как идеи, а также голосования за идеи, работают путем отправки приложением сообщения определенного формата на сервер. Сервер, получив сообщение, делает соответствующее изменение в базе данных и отправляет сообщение об этом всем участникам комнаты. Приложения участников, получив такое сообщение от сервера, делают изменения в локальной базе данных, тем самым обновляя их в реальном времени.

### Просмотр идей

Просмотр идей в комнате происходит с помощью запросов к локальной базе данных приложения и не требует подключения к серверу.

### Выход из комнаты

Данный функционал доступен только участнику мозгового штурма.

При попытке покинуть комнату происходит API запрос на сервер. В случае успеха, пользователь удаляется из таблицы участников комнаты на сервере и переходит в раздел «История мозговых штурмов».

### Удаление комнаты

Данный функционал доступен только создателю комнаты.

При попытке удалить комнату происходит API запрос на сервер. В случае успеха, комната удаляется из базы данных сервера и сервер сообщает об удалении всем участникам комнаты с помощью WebSocket соединения. Все пользователи переходят к разделу «История мозговых штурмов».

### Завершение мозгового штурма

Данный функционал доступен только создателю комнаты.

При попытке завершить мозговой штурм происходит API запрос на сервер. В случае успеха, сервер собирает информацию о мозговом штурме. Сервер получает тему мозгового штурма из поля details, получается все сообщения, помеченные идеями, и сортирует их по идеям и количеству голосов. Идеи, имеющие большее количество голосов, находятся выше. Сервер форматирует эти данные в формат HTML для удобного их отображения в приложении при просмотре идей. После форматирования создается новая запись о завершенном мозговом штурме в базе данных, а комната, в которой мозговой штурм завершился, удаляется. Далее все участники мозгового штурма уведомляются о том, что мозговой штурм завершен и переходят в раздел «История мозговых штурмов».

### Обмен идей в формате Round Robin

Пользователь начинает работу с идеями, начиная с идеи, которая была ему назначена при создании комнаты (см. п. «Создание виртуальной комнаты для мозгового штурма»). После того, как пользователь отправил своё дополнение к идее с помощью WebSocket сообщения, он добавляется в список пользователей, которые закончили работу над идеями. После этого сообщения другие сообщения о дополнении идеи пользователем игнорируются, до начала следующего раунда. Когда список пользователей, которые закончили работу над идеями, совпадет со списком участников комнаты, каждый участник получит новую идею (идею другого пользователя) для работы с помощью WebSocket сообщения, список пользователей, которые закончили работу над идеями, очиститься, а значение поля *completed\_rounds* увеличится на один. Как только число *completed\_rounds* совпадет с количеством участников, этап «Round Robin» считается завершенным. Тогда информация Round Robin Data удаляется из базы данных сервера (наличие этой информации используется сервером для идентификации процесса обмена идей Round Robin), а все участники уведомляются о завершении этого этапа с помощью WebSocket сообщения и переходят к разделу «Чат классического мозгового штурма».

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

Входные данные представлены в виде нажатий на экран мобильного устройства, выполняемых в процессе взаимодействия с приложением. Кроме того, входными данными также является ввод текста с клавиатуры мобильного устройства в специально предназначенные для этого текстовые поля. Выбор такого формата входных обусловлен тем, что это самый основной способ сбора входных данных в мобильных приложениях.

Входными данными для сервера являются API-запросы и WebSocket сообщения в формате JSON. JSON представляет собой удобный формат для обмена информацией, поскольку существует множество решений, которые позволяют переводить данные в JSON и обратно. Более того, JSON также удобен для восприятия непосредственно человеком, что позволяет взаимодействовать с сервером даже без интерфейса приложения. В свою очередь, API-запросы позволяют пользователю передавать команды серверу в удобном формате, а WebSocket соединение реализует двухканальную связь, позволяя обмениваться данными в реальном времени.

В качестве выходных данных служат преимущественно текстовые данные, а также отчёт мозгового штурма в формате pdf. Pdf является популярным кросс-платформенным форматом, который будет открываться одинаково вне зависимости от операционной системы. Таким образом, отчёт мозгового штурма после экспортирования можно просматривать на любом устройстве, которое поддерживает этот формат, даже если оно не на базе Android.

Выходными данными сервера являются ответы сервера в виде JSON. Преимущества и обоснование выбора этого формата данных описано выше.

## Описание и обоснование выбора состава технических и программных средств

*Минимальные требования к клиентскому оборудованию (мобильному устройству):*

* Android 8.0 или выше;
* 500 МБ свободной оперативной памяти;
* 100 МБ свободного места на хранилище устройства;
* Одноядерный процессор с частотой 1 ГГц;
* Возможность выхода в интернет;

*Минимальные требования к серверному оборудованию:*

* Intel Xeon E5-2620 V3 или новее;
* 2 ГБ оперативной памяти;
* 2 ГБ SSD хранилища;
* Установленные средства для корректной работы используемого Django фреймворка;
* Установленные средства для работы с PostgreSQL или отдельный сервер с базой данных.

Оборудование, которое не имеет установленных требуемых программных средств, не сможет запустить приложение или работать с ним. При этом устройства могут иметь технические характеристики хуже, чем те, которые указаны в минимальных требованиях, однако тогда пользователь может столкнутся с медленной работой приложения или отказа некоторых функций. Устройства, соответствующие минимальным требованиям, позволяют пользователю получить бесперебойный и полный доступ к функционалу приложения.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Ориентировочная экономическая эффективность

Продукт на данном этапе не подразумевает наличие монетизации.

## Предполагаемая потребность

Мобильное приложение «StormMaster» может быть полезно людям почти любого возраста. Предполагается, что приложение будет использоваться людьми, которые хотят придумать решение какой-то проблемы совместными усилиями.

* 1. **Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами**

«StormMaster» имеет достаточный основной и дополнительный функционал, чтобы конкурировать на рынке похожих приложений. В таблице № 1 представлен сравнительный анализ функционала.

*Таблица № 1 – Сравнительный анализ функционала*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Критерий** | **Аналоги** | | | | |
| StormMaster | Miro | MindMeister | StormBoard | EdrawMind |
| Android версия приложения | + | + | + | + | + |
| Android версия приложения реализует основной функционал (например, по сравнению с Web версией, если такая присутствует) | + | + | - | - | + |
| Стабильность работы Android приложения | + | + | + | - | + |
| Возможность регистрации пользователя | + | + | + | + | + |
| Пользователь может пользоваться приложением в качестве читателя/зрителя без регистрации | - | + | + | - | + |
| Встроенная возможность голосования за идеи | + | - | - | + | - |
| Создание комнат для сессий | + | + | + | + | + |
| Возможность выгнать пользователя из комнаты | - | + | + | + | + |
| Преднастройка пространства для конкретной техники мозгового штурма | + | + | + | + | - |
| Автоматизация процесса проведения мозгового штурма | + | - | - | - | - |
| Чат для участников в реальном времени | + | - | - | + | - |
| Возможность комментировать чужие идеи | + | + | + | + | + |
| Возможность автоматического формирования текстового отчёта мозгового штурма | + | - | + | + | + |
| Возможность экспортирование материалов мозгового штурма в формате pdf | + | + | + | + | + |
| Сохранение историй сессий | + | + | + | + | + |
| Наличие визуальных инструментов (досок, схем) | - | + | + | + | + |
| Работа в режиме оффлайн (просмотр уже созданных материалов) | + | - | + | - | + |

В рамках сравнительного анализа удалось убедится в том, что заявленный функционал приложения отвечает базовым потребностям пользователя.

Также удалось выделить главного конкурента – Stormboard. Несмотря на то, что он имеет большее количество минусов по сравнению с другими конкурентами, его функционал максимально приближен к разрабатываемому приложению: есть возможность создавать комнаты для сессий, использовать преднастройки для разных видов мозгового штурма, голосовать за идеи, комментировать их, общаться в реальном времени и экспортировать информацию о проведенном мозговом штурме в формате pdf, а также сохранять историю сессий.

Однако, несмотря на схожесть, разрабатываемое приложение имеет ряд преимуществ перед StormBoard:

* **Android версия приложения.** В целом, у StormBoard заявлено Android версия приложения и её даже можно скачать из Google Play, однако она работает крайне нестабильно. Во-первых, оно не всегда открывается с первого раза. Для тестирования работы приложения пришлось предпринимать попытки его открыть в течении дня. Во-вторых, в приложении не работает какой-либо ввод с клавиатуры Android устройства. Из-за этого у пользователя теряется возможность использовать чат, комментировать какие-либо идеи, а также добавлять какие-то новые идеи в процессе мозгового штурма.

Разрабатываемое приложение StormMaster предоставит пользователю полный доступ к функционалу, а также будет работать без подобных сбоев.

* **Автоматизация проведения мозгового штурма.** StormBoard, как и всех других конкурентов, объединяет одно – они дают инструменты для проведения мозгового штурма, однако конкуренты не помогают его организовывать, инструменты нужно использовать вручную. Например, техника Round Robin подразумевает то, что идеи будут передаваться по кругу. StormBoard, в целом, позволяет организовать подобную технику, но все предложенные идеи должны передаваться между пользователями вручную.

Разрабатываемое приложение StormMaster автоматизирует реализуемые техники. Например, в упомянутой выше Round Robin идеи будут распределяться автоматически.

* **Работа в офлайн режиме.** StormBoard не поддерживает какую-либо работу без интернета. Приложение не может открыться без доступа к нему.

Разрабатываемое приложение StormMaster позволит пользователю просматривать отчёты о уже проведенных мозговых штурмах без необходимости подключения к Интернету.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19.101-77 Виды программ и программных документов // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
2. ГОСТ 19.102-77 Стадии разработки // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
3. ГОСТ 19.103-77 Обозначения программ и программных документов // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
4. ГОСТ 19.104-78 Основные надписи // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
5. ГОСТ 19.105-78 Общие требования к программным документам // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
8. ГОСТ 19.603-78 Общие правила внесения изменений // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
10. ГОСТ 19.301-78 Программа и методика испытаний // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
11. ГОСТ 19.401-78 Текст программы // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
12. ГОСТ 19.404-79 Пояснительная записка // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
13. ГОСТ 19.505-79 Руководство оператора // официальное издание Единая система программной документации: Сборник национальных стандартов. - М.: Стандартинформ, 2010
14. Square Open Source. Retrofit: A type-safe HTTP client for Android and Java [Электронный ресурс]. URL: [https://square.github.io/retrofit/](https://square.github.io/retrofit/%20) (дата обращения: 01.04.2025)
15. Android Developers. Room Persistence Library [Электронный ресурс]. URL: [https://developer.android.com/jetpack/androidx/releases/room](https://developer.android.com/jetpack/androidx/releases/room%20) (дата обращения: 01.04.2025)
16. iText Software. iText 7. [Электронный ресурс]. URL: <https://itextpdf.com/> (дата обращения: 01.04.2025)
17. Square Open Source. OkHttp [Электронный ресурс]. URL: <https://square.github.io/okhttp/> (дата обращения: 01.04.2025)
18. Google Developers. Material Design Components for Android [Электронный ресурс]. URL: <https://material.io/develop/android>(дата обращения: 01.04.2025)
19. Django Software Foundation. Django Documentation [Электронный ресурс]. URL: <https://docs.djangoproject.com/> (дата обращения: 01.04.2025)
20. Django REST framework. Official Documentation [Электронный ресурс]. URL: <https://www.django-rest-framework.org/> (дата обращения: 01.04.2025)
21. Djoser. REST implementation of Django auth [Электронный ресурс]. URL: <https://djoser.readthedocs.io/> (дата обращения: 01.04.2025)
22. DRF-YASG. Swagger/OpenAPI for DRF [Электронный ресурс]. URL: <https://drf-yasg.readthedocs.io/> (дата обращения: 01.04.2025)
23. Django Channels. [Электронный ресурс]. URL: <https://channels.readthedocs.io/> (дата обращения: 01.04.2025)
24. Django Project. Daphne: Django Channels HTTP/WebSocket server [Электронный ресурс]. URL: <https://github.com/django/daphne> (дата обращения: 01.04.2025)

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводи- тельного документа и дата | Подпись | Дата |
| изменен-ных | заме-  ненных | новых | аннули-  рован-  ных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |