|  |  |
| --- | --- |
| **­­** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***Создание распределенной информационной\_\_\_\_\_\_\_ системы в соответствии с вариантом\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ5-63Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_К. А. Буйдина\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_А.И. Канев\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2025 г.*

|  |  |
| --- | --- |
| **­­** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***Создание распределенной информационной\_\_\_\_\_\_\_ системы в соответствии с вариантом\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ5-63Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_А. А. Смирнов\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_А.И. Канев\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2025 г.*

|  |  |
| --- | --- |
| **­­** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***Создание распределенной информационной\_\_\_\_\_\_\_ системы в соответствии с вариантом\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ5-63Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_И. Д. Хабленко\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_А.И. Канев\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2025 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_ИУ5\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_В.И. Терехов\_\_

(И.О.Фамилия)

«\_09\_» \_\_\_\_февраля\_\_\_\_\_\_\_ 2025 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине \_\_\_\_Сетевые технологии в АСОИУ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_ИУ5-63Б\_\_\_Буйдина К. А., Смирнов А. А., Хабленко И. Д.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Тема курсовой работы Распределенная информационная система обмена сообщениями\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_в реальном времени\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Направленность КР (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_УЧЕБНАЯ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_КАФЕДРА\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Задание*** Разработать автоматизированную распределенную систему для обмена сообщениями в\_\_\_\_ реальном времени\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление курсовой работы:***

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_09\_» \_\_\_\_февраля\_\_\_\_\_\_\_ 2025 г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_А.И. Канев\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ К. А. Буйдина \_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ А. А. Смирнов\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ И. Д. Хабленко\_\_\_

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**КАЛЕНДАРНЫЙ ПЛАН**

**на выполнение курсовой работы**

|  |  |  |
| --- | --- | --- |
| по дисциплине | Сетевые технологии в АСОИУ | |
| Студент группы | ИУ5-63Б Буйдина К. А., Смирнов А. А., Хабленко И. Д. | |
|  | (Фамилия, имя, отчество) | |
| Тема курсовой работы | | Распределенная информационная система обмена |
| сообщениями в реальном времени | | |
|  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Наименование этапов выпускной квалификационной работы** | **Сроки выполнения этапов** | | **Отметка о выполнении** | |
| **план** | **факт** | **Руководитель КР** | **Куратор** |
| 1. | Выбор темы; формирование команды и ТЗ | *\_22.02.25\_*  *Планируемая дата* | *22.02.24* |  |  |
| 2. | Создание макета Figma, диаграммы последовательности и формирование swagger | *29.03.25\_*  *Планируемая дата* | *29.03.24* |  |  |
| 3. | Оформление РПЗ, ПМИ, РСА, РП | *27.04.25\_*  *Планируемая дата* | *27.04.24* |  |  |
| 4. | Тестирование распределенной системы и подготовка проекта к демонстрации проектапрезеннеобходимости) | *15.05.25\_*  *Планируемая дата* | 08.05.2025 |  |  |
| 5. | Защита курсовой работы | *16.05.25\_*  *Планируемая дата* | 12.05.2025 |  |  |

Студент *\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Руководитель работы *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись, дата) (подпись, дата)

Студент *\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

Студент *\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

# Содержание

[**Содержание 6**](#_Toc197952340)

[**ВВЕДЕНИЕ 7**](#_Toc197952341)

[**ТРАНСПОРТНЫЙ УРОВЕНЬ 8**](#_Toc197952342)

[**КАНАЛЬНЫЙ УРОВЕНЬ 11**](#_Toc197952343)

[**ПРИКЛАДНОЙ УРОВЕНЬ 14**](#_Toc197952344)

[**ЗАКЛЮЧЕНИЕ 20**](#_Toc197952345)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21**](#_Toc197952346)

[**ПРИЛОЖЕНИЕ А ТЕХНИЧЕСКОЕ ЗАДАНИЕ 22**](#_Toc197952347)

[**ПРИЛОЖЕНИЕ Б ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 28**](#_Toc197952348)

[**ПРИЛОЖЕНИЕ В РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 33**](#_Toc197952354)

[**ПРИЛОЖЕНИЕ Г РУКОВОДСТВО СИСТЕМНОГО АДМИНИСТРАТОРА 39**](#_Toc197952355)

# ВВЕДЕНИЕ

В современном мире обмен сообщениями и файлами в реальном времени является неотъемлемой частью коммуникации как в повседневной жизни, так и в бизнес-среде. Для обеспечения надежной и эффективной передачи данных необходимо учитывать возможные помехи в канале связи, а также корректную сборку и доставку информации.

Данный проект представляет собой трехуровневую систему обмена текстовыми сообщениями и файлами. Прикладной уровень – интерфейс чата с возможностью отправки и получения данных. Транспортный уровень – разбиение сообщений на сегменты и их сборка. Канальный уровень – эмуляция передачи данных по ненадежному каналу связи с помехами. Каждый уровень реализован в виде отдельного веб-сервиса, что обеспечивает модульность и масштабируемость системы.

Такое разделение функционала позволяет упростить разработку, тестирование и дальнейшее расширение системы. В результате будет создано решение, способное эффективно работать в условиях реального времени, обеспечивая пользователей удобным и безопасным способом обмена сообщениями и файлами.

Нефункциональные требования к разрабатываемой системе:

* 1. Интерфейс системы и текст ошибок должны быть русифицируемы.

В ходе работы необходимо выполнить следующие задачи:

* 1. Разработать дизайн приложения.
  2. Реализовать прикладной уровень и интерфейс.
  3. Реализовать транспортный уровень
  4. Реализовать канальный уровень
  5. Подготовить набор документации, включающий РПЗ, ТЗ и набор диаграмм.

# ТРАНСПОРТНЫЙ УРОВЕНЬ

Транспортный уровень предназначен для доставки данных и представляет из себя механизм передачи сообщений. Он также является связующим между прикладным и канальным уровнями. На рисунках 1 и 2 представлены диаграммы последовательности и развёртывания, соответственно. На них можно увидеть, что транспортный уровень должен выдерживать высокую нагрузку, чтобы обеспечивать связь двух бэкендов и не становиться «узким горлышком».

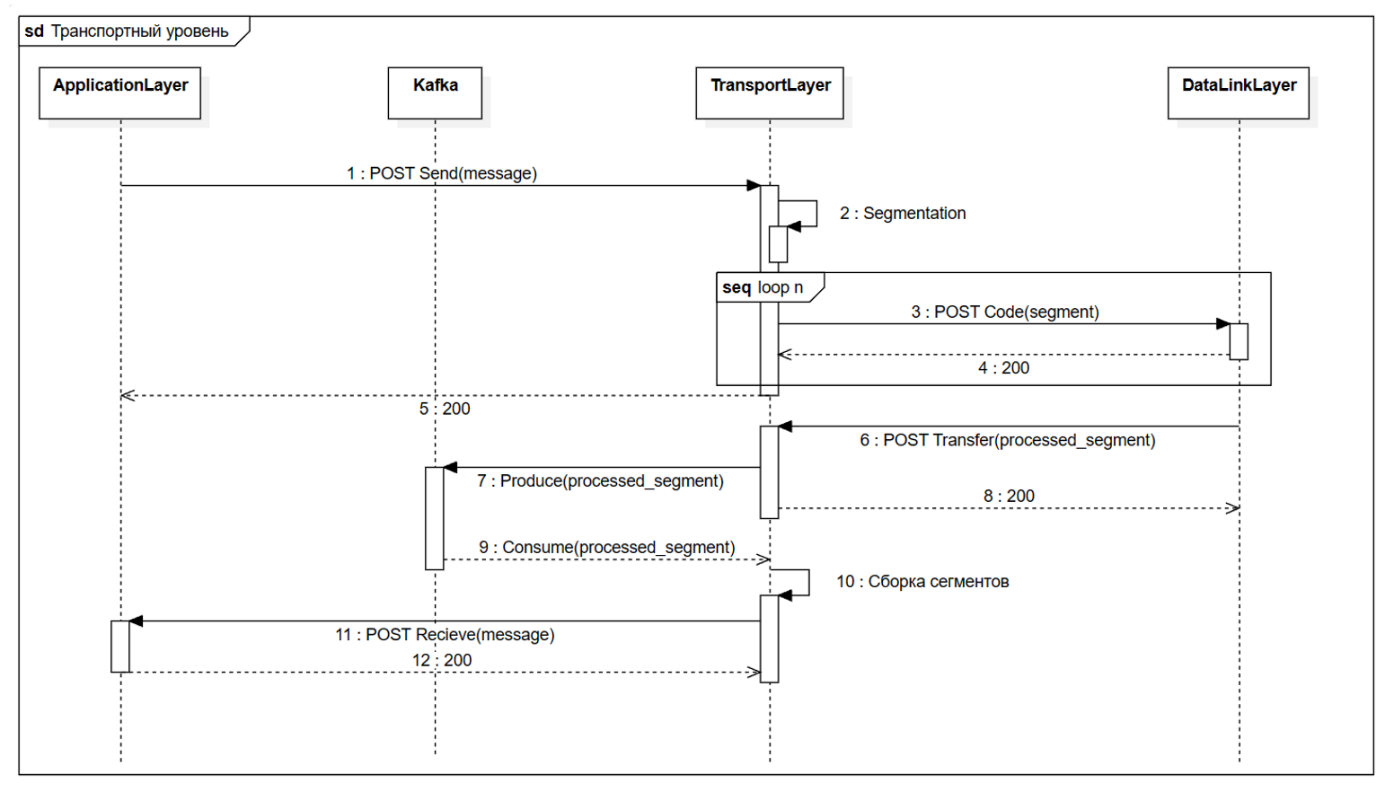


Рисунок 1 – Диаграмма последовательности транспортного уровня

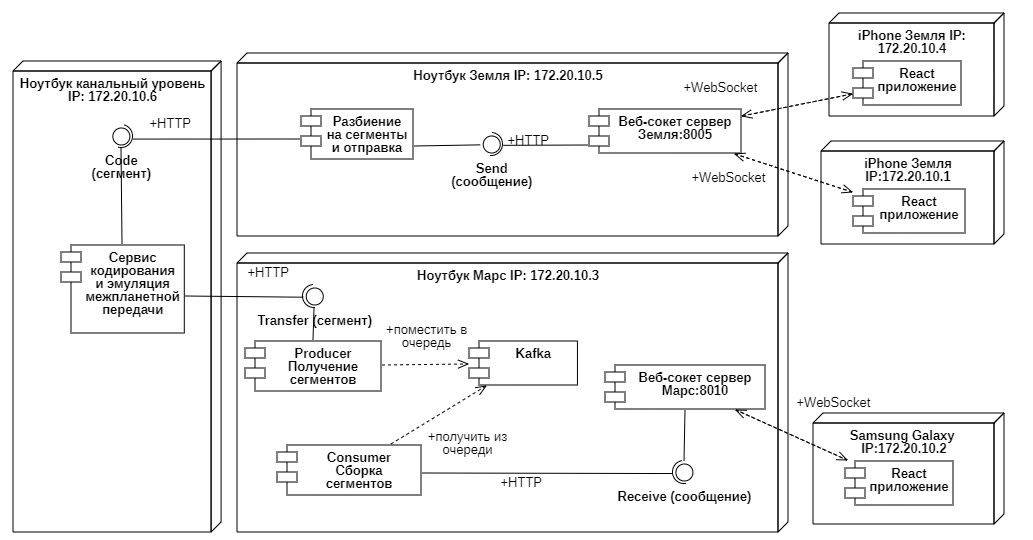


Рисунок 2 – Диаграмма развёртывания

Рассмотрим алгоритм работы транспортного уровня более подробно. Отправной точкой является вызов метода Send прикладным уровнем, в теле запроса в метод передаются имя отправителя, содержание сообщения и метка времени отправки этого сообщения. Далее сообщение сегментируется по 100 байт, где каждый сегмент имеет свой номер, имя пользователя, чье сообщение было сегментированг, время отправки и содержание сегмента, и посегментно отправляется на канальный уровень посредством вызова метода Code. Обработав сегменты, канальный уровень возвращает каждый из них, вызывая метод Transfer. При выявлении ошибки в теле запроса может быть передан флаг ошибки, чтобы указать конечному пользователю, что сообщение содержит ошибку и не может быть прочитано.

В текущей реализации транспортного уровня используется механизм очереди отправки пакета. Полученный с канального уровня сегмент помещается в Kafka с учетом сохранения правильного порядка сегментов. Kafka – это распределенный брокер сообщений, представляющий собой отказоустойчивые конвейеры, работающие по принципу «публикация/подписка» и позволяющие обрабатывать потоки событий [5]. Из очереди раз в 2 секунды они собираются в единое сообщение и передаются на прикладной уровень с вызовом метода Receive. Сообщение содержит имя пользователя, текст сообщения, время отправки. В случае, когда за 2 цикла часть сегментов не была принята, то сообщение передаётся на прикладной уровень с признаком ошибки. Если ошибки нет, то это поле возвращается пустым.

Методы, реализуемые транспортным уровнем, представлены в таблице 1:

**Таблица 1 – Методы транспортного уровня**

|  |  |  |
| --- | --- | --- |
| Метод | Входные параметры | Описание |
| POST Send | id integer  time string  sender string  payload string | Отправка сообщения с прикладного уровня на транспортный |
| POST Transfer | time string  payload string  number\_of\_segments integer  segment\_number integer  sender string | Отправка декодированного сегмента с канального уровня на транспортный |

# КАНАЛЬНЫЙ УРОВЕНЬ

Данный уровень эмитирует взаимодействие с удаленным сетевым узлом через канал с помехами. Передаваемая информация защищается циклическим кодом [7,4]. Полученный от транспортного уровня json сегмента кодируется в битовый формат соответствующим кодом, вносятся ошибки и затем декодируется для последующей отправки обратно на транспортный уровень. Архитектурно решение состоит из серверной части, реализованной на Node.js, работающей на порту 3050 с единственным эндпоинтом /code, принимающим POST-запросы с JSON-данными. Для обработки данных разработан специальный программный пакет, включающий кодировщик/декодировщик циклического кода [7,4], функции для работы с битовыми последовательностями и эмулятор канала с помехами.

Канальный уровень получает от транспортного сегмент размером 100 байт, вероятность внесения ошибки составляет 10%, а вероятность потери кадра 2%.

Циклический код — линейный, блочный код, обладающий свойством цикличности, то есть каждая циклическая перестановка кодового слова также является кодовым словом. Используется для преобразования информации для защиты её от ошибок. [6] Кодирование и декодирование выполняются с использованием полиномиальной арифметики. Сообщение представляется в виде полинома, а код строится путем деления на порождающий полином. Остаток от деления используется для формирования контрольной суммы.

Данная реализация обеспечивает надежную передачу данных через ненадежный канал связи, эффективно используя возможности циклического кодирования для защиты от ошибок и сохраняя целостность передаваемой информации даже в условиях помех и вероятностных потерь кадров.

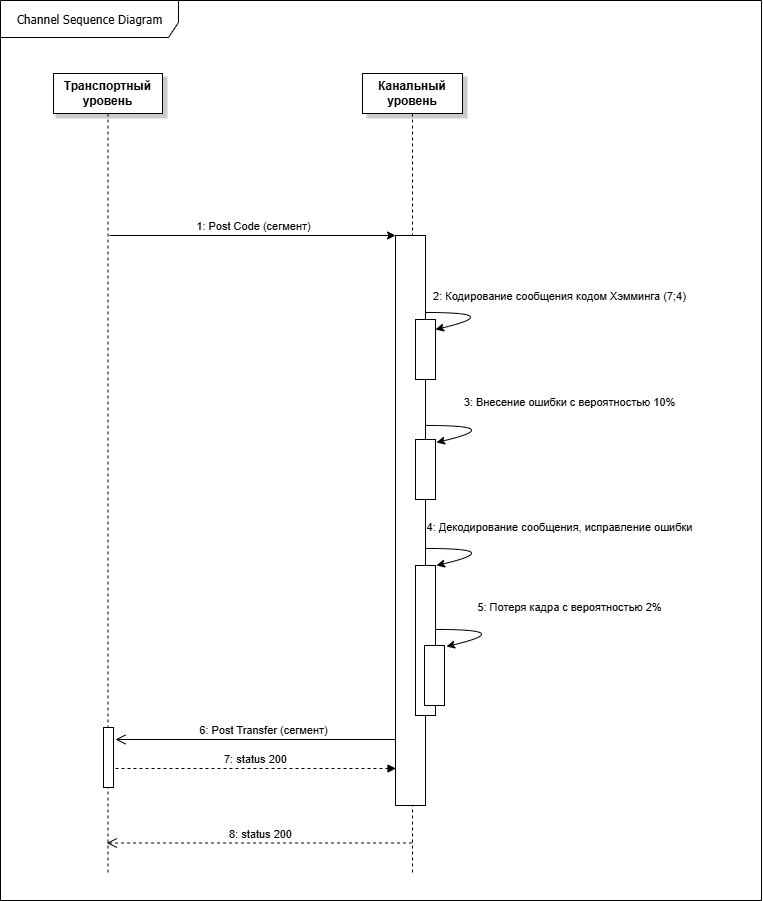
****

Рисунок 3 – Диаграмма последовательности канального уровня

Метод, используемый на канальном уровне (Таблица 2):

**Таблица 2 – Методы канального уровня**

| Метод | Входные параметры | Описание |
| --- | --- | --- |
| POST Code | message\_id string  payload string  number\_of\_segments integer  sequence\_number integer | Полученный от транспортного уровня json сегмента кодируется циклическим кодом [7, 4] в битовый формат. Вносится ошибка в 1 бит сегмента. Далее сегмент декодируется и, если исправляется ошибка, либо же сегмент теряется. Затем сегмент передается на транспортный уровень. |

# ПРИКЛАДНОЙ УРОВЕНЬ

Прикладной уровень предназначен для отправки и приема данных. Он также является инструментом взаимодействия пользователя с системой и отображения всех данных, приходящих в реальном времени. Для подключения необходимо ввести имя пользователя, после чего выполняется установка WebSocket соединения с соответствующим сервером - Земля или Марс. Чат является общим и сохраняет историю сообщений. Пользователи Земли получают сообщения моментально, а пользователи Марса получают сообщение после того, как оно проходит через транспортный и канальный уровень. Если сообщение доставляется с признаком ошибки, то в чате отображается значок ошибки.

Для реализации пользовательского интерфейса прикладного уровня используется библиотека React [1] с менеджером состояний Redux Toolkit [2] и с библиотекой для запросов Axios. В качестве UIkit взят MUI [3], а за основу дизайна выбран сайт https://futureengineer.co.uk/expertise/aerospace-engineering-recruitment-agency. Для сервера используется Node.js [4] с фреймворком Express.js.

WebSocket — это протокол связи, который обеспечивает непрерывное и двустороннее соединение между клиентом и сервером через одно TCP-соединение. Он позволяет обмениваться данными в режиме реального времени, что делает его идеальным выбором для чат-приложений. В чатах WebSocket используется для обеспечения мгновенной доставки сообщений между пользователями. При установлении соединения между клиентом и сервером через WebSocket, клиент и сервер могут отправлять и принимать сообщения в любое время без необходимости постоянного обновления страницы или выполнения дополнительных запросов. При использовании WebSocket в чат-приложениях, каждое сообщение, отправленное одним пользователем, мгновенно передается всем остальным участникам чата. Это позволяет пользователям видеть сообщения в режиме реального времени без необходимости обновления страницы или ожидания новых данных от сервера.

Благодаря непрерывному соединению WebSocket, чат-приложения могут предоставлять такие функции, как моментальная доставка сообщений, индикация онлайн-статуса других пользователей, а также уведомления о новых сообщениях. Это делает пользовательский опыт более интерактивным и удобным для всех участников чата. Кроме того, WebSocket обеспечивает эффективное использование ресурсов сервера, поскольку он позволяет установить одно постоянное соединение для передачи всех сообщений, вместо создания отдельного HTTP-запроса для каждого нового сообщения. Это снижает нагрузку на сервер и уменьшает задержки при передаче сообщений.

В целом, использование WebSocket в чат-приложениях обеспечивает быструю и надежную передачу сообщений в реальном времени, что делает его основным инструментом для создания современных интерактивных чат-систем.

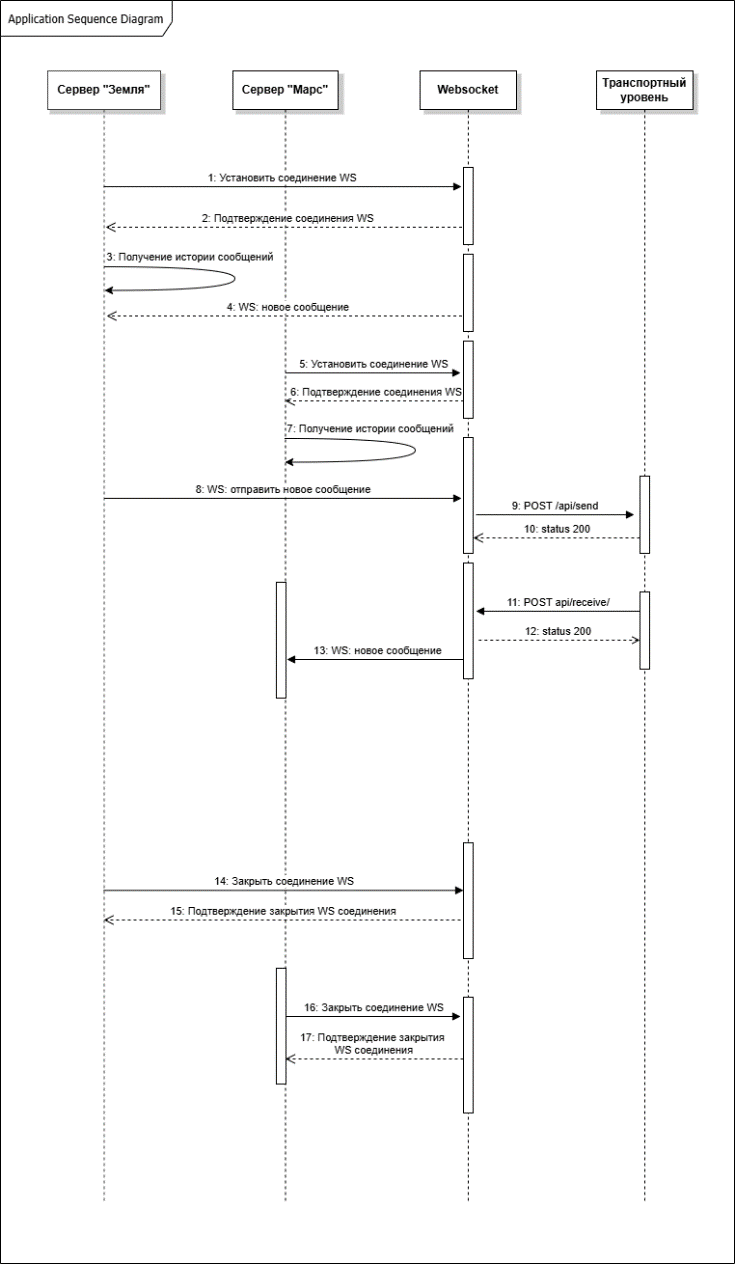
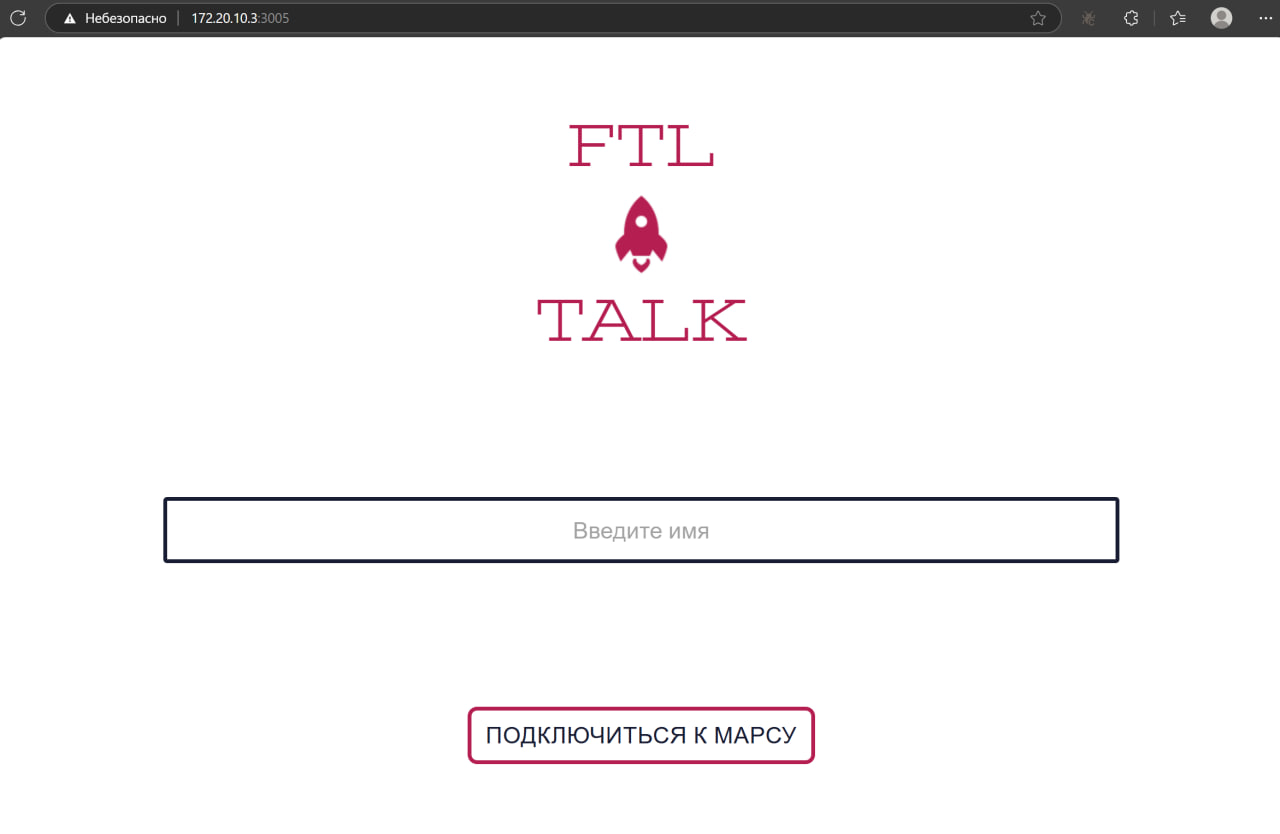


Рисунок 4 – Диаграмма последовательности прикладного уровня

**Таблица 3 – Методы прикладного уровня**

|  |  |  |
| --- | --- | --- |
| Метод | Входные параметры | Описание |
| POST Receive | user integer  payload string  time string  error boolean | Прием декодированного сегмента с транспортного уровня на прикладной |



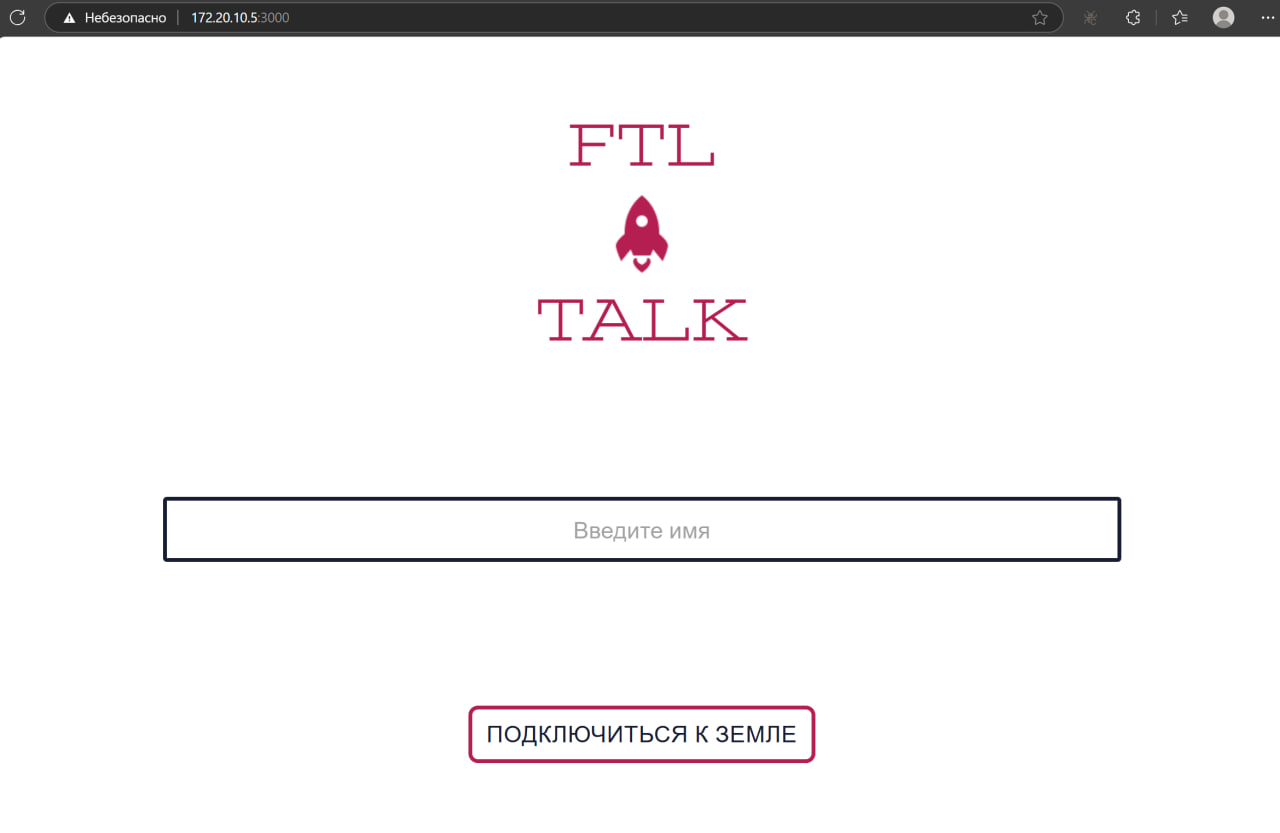


Рисунок 5 – Скриншот работы программы (авторизация): десктопная версия

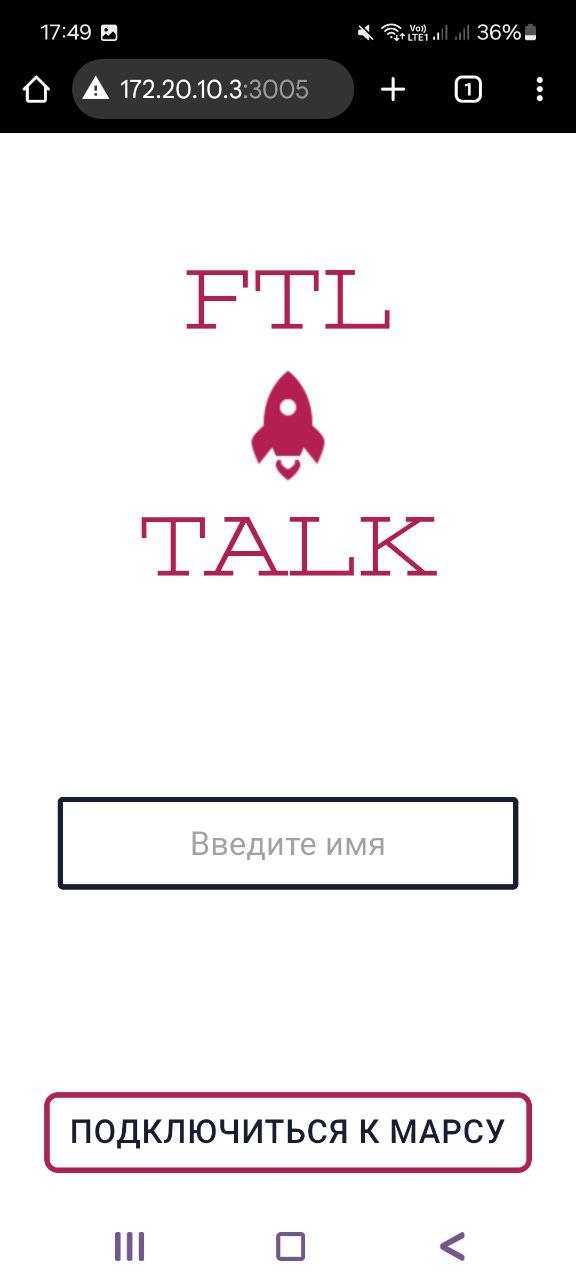
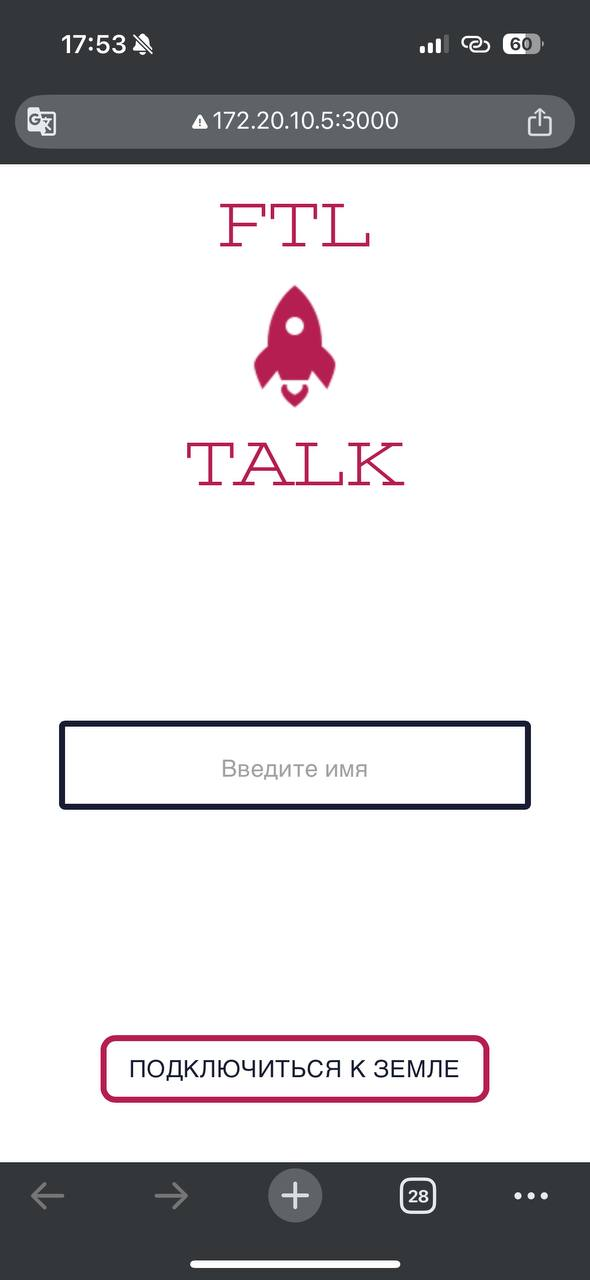
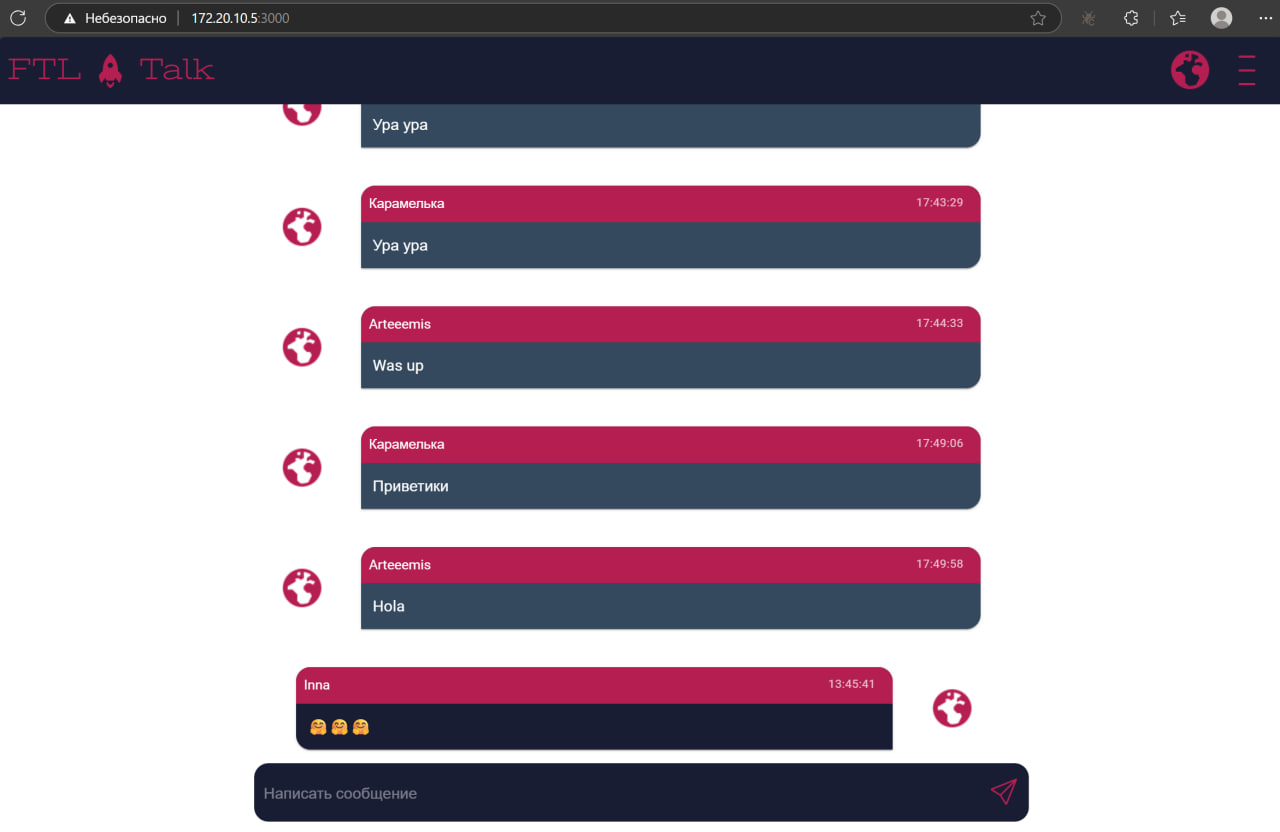
 

Рисунок 6 – Скриншот работы программы (авторизация): мобильная версия 

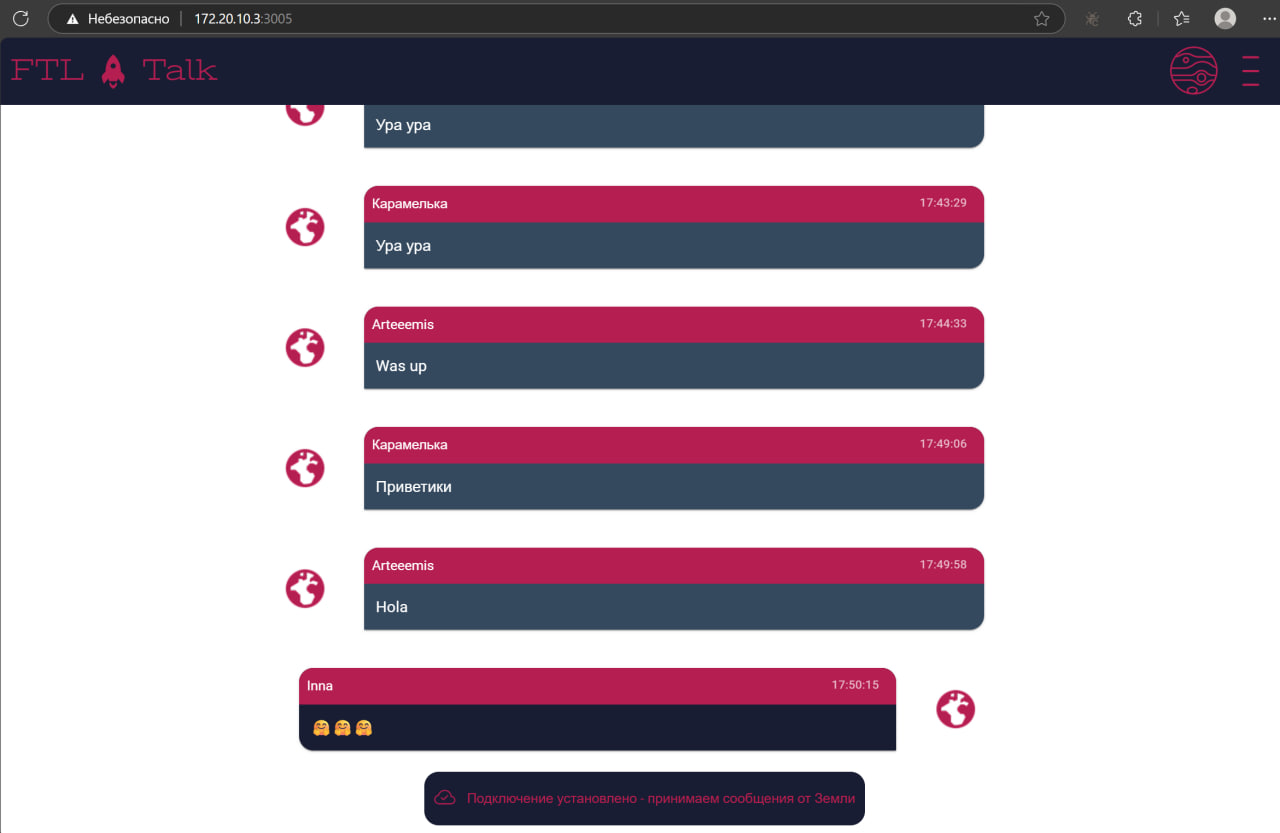


Рисунок 7 – Скриншот чата: отправка сообщение: экран с Земли и с Марса

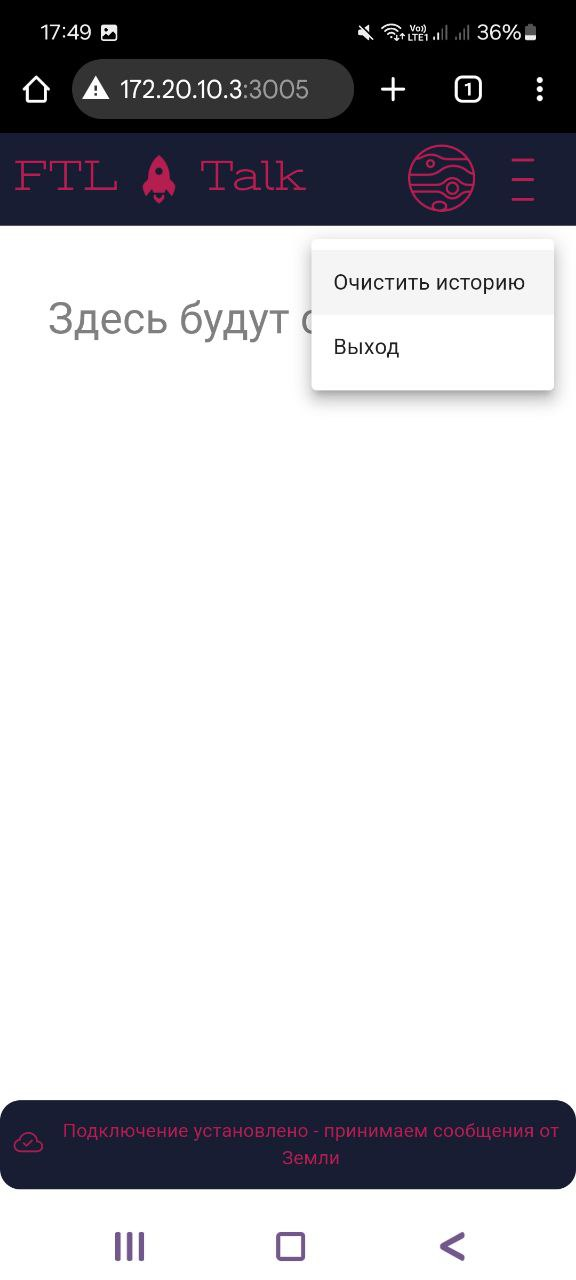
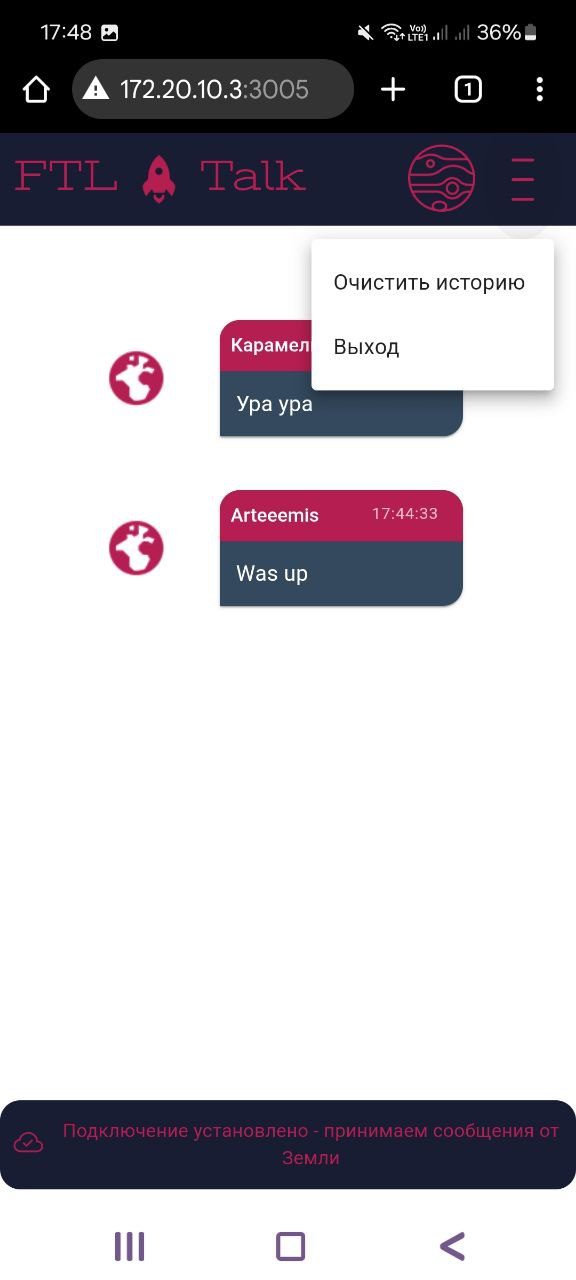


Рисунок 8 – Скриншот чата: пользователь очищает историю: экран Марса

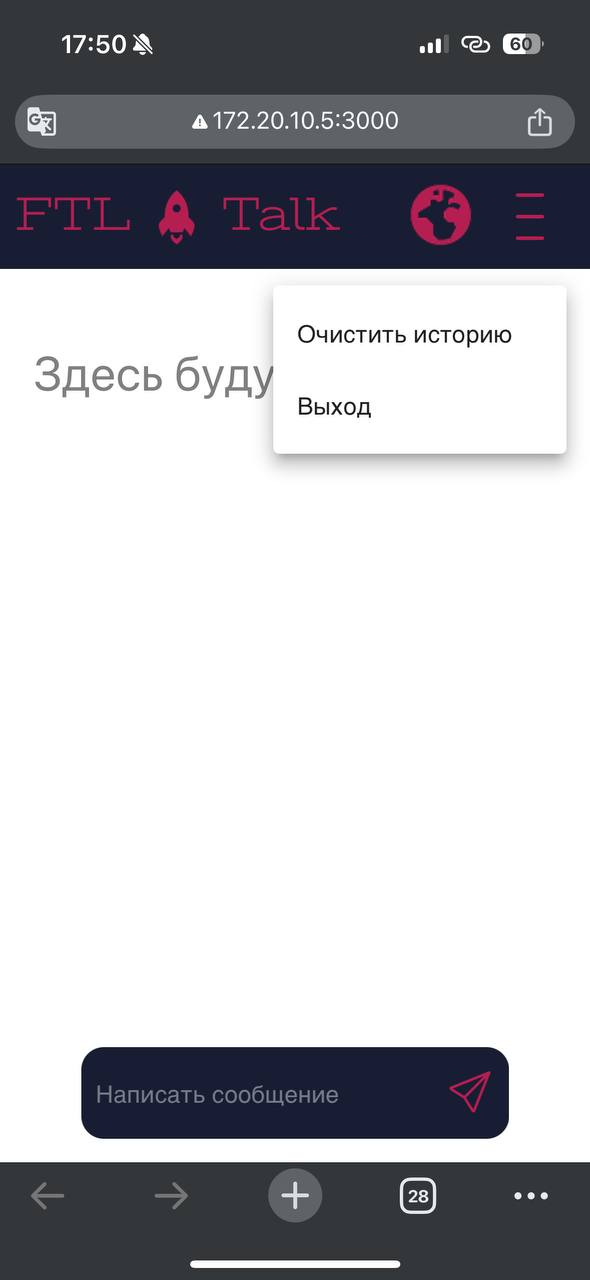
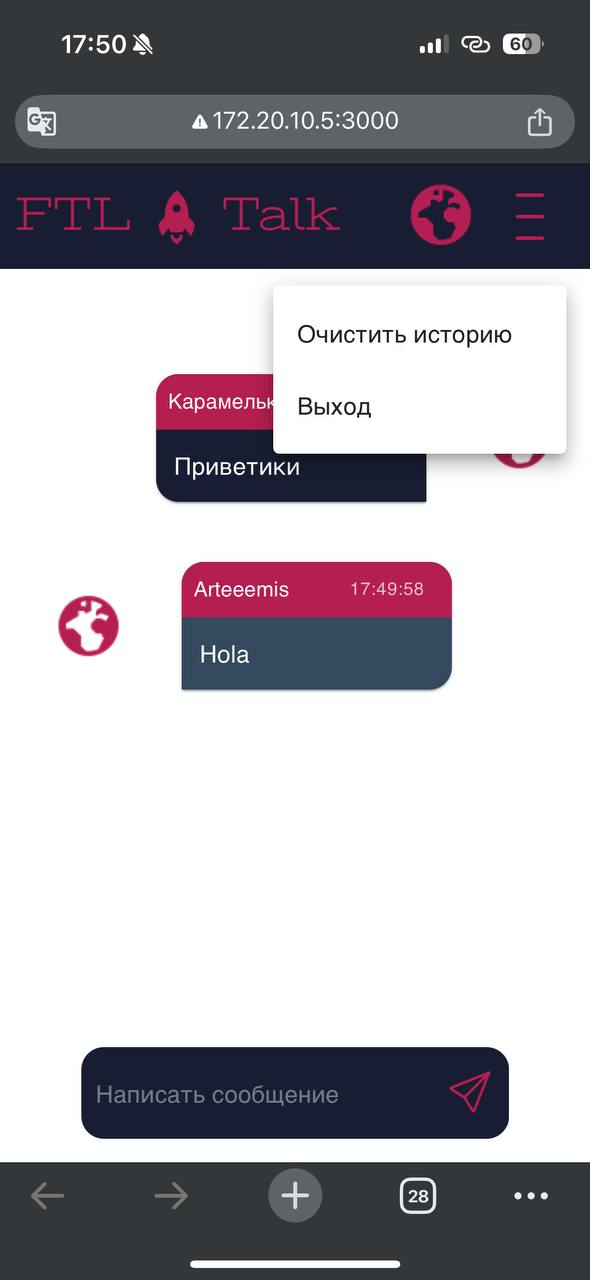


Рисунок 9 – Скриншот чата: пользователь очищает историю: экран Земля

# ЗАКЛЮЧЕНИЕ

В ходе работы были достигнуты следующие результаты:

1. Был разработан дизайн приложения с помощью набора стилей CSS и HTML тегов.
2. Были разработаны прикладной, транспортный и канальный уровни.
3. Разработан интерфейс с использованием технологии React Framework и подключен к веб-сервису.
4. Методы прикладного, транспортного и канального уровней задокументированы через Swagger.
5. Подготовлен набор документации, включающий РПЗ, ТЗ и набор UML диаграмм.
6. Исходный код проекта доступен в GitHub.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Руководство по React [Электронный ресурс] // React. URL: https://ru.legacy.reactjs.org/ (дата обращения: 20.04.2025).
  2. Руководство по Redux [Электронный ресурс] // React-redux. URL: https://ru.react-redux.js.org/tutorials/quick-start/ (дата обращения: 20.04.2025).
  3. Руководство по MUI [Электронный ресурс] // MUI. URL: https://mui.com/material-ui/getting-started/ (дата обращения: 20.04.2025).
  4. ws: a Node.js WebSocket library: документация [Электронный ресурс] // Github. URL: https://github.com/websockets/ws (дата обращения: 20.04.2025).
  5. Apache Kafka: обзор [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/piter/articles/352978/ (дата обращения: 20.04.2025).
  6. Циклический код. [Электронный ресурс] // Рувики. URL: https://ru.ruwiki.ru/wiki/ (дата обращения: 20.04.2025).

# ПРИЛОЖЕНИЕ А ТЕХНИЧЕСКОЕ ЗАДАНИЕ

|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Утверждаю

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_2025г.

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Дисциплина «Сетевые технологии в АСОИУ»

Техническое задание

Вариант 25

Студенты группы ИУ5-63Б:

Буйдина К. А.

Смирнов А. А.

Хабленко И. Д.

2025г.

1. **Наименование:**

Распределённая информационная система обмена текстовыми сообщениями в реальном времени.

1. **Основание для разработки:**

Основанием для разработки является учебный план МГТУ им. Баумана кафедры ИУ5 на 6 семестр.

1. **Исполнители:**

Буйдина К. А. (транспортный уровень) – группа ИУ5-63Б

Смирнов А. А. (прикладной уровень) – группа ИУ5-63Б

Хабленко И. Д. (канальный уровень) – группа ИУ5-63Б

1. **Цель разработки:**

Разработать распределённую систему для обмена текстовыми сообщениями в реальном времени, состоящую из трех уровней: прикладной, транспортный и канальный, каждый из которых реализуется отдельным веб-сервисом, разбиваясь на сегменты по 100 байт и кодируясь [7,4] циклическим кодом , и доходит до получателя с вероятностью ошибки в 10% и вероятностью потери 2%. Интерфейс пользователя соответствует дизайну сайта https://futureengineer.co.uk/expertise/aerospace-engineering-recruitment-agency/.

1. **Функциональные требования**
   1. Прикладной уровень:
      1. Страница приложения с окном чата для отправки и просмотра полученных текстовых сообщений с указанием отправителя и времени отправки;
         1. При подключении к чату пользователь должен ввести имя, которое будет передаваться с каждым новым сообщением, выбрать сервер и войти с помощью кнопки Войти;
         2. После входа появляется возможность отправки сообщений по установленному WebSocket соединению, используя кнопки «Добавить сообщение», «Отправить»;
         3. По кнопке «Выйти» чат и логин очищаются, а ws-подключение закрывается;
         4. В случае, если сообщение пришло с признаком ошибки, сообщение не отображается, а вместо него у получателей появляется соответствующее сообщение об ошибке;
         5. Дизайн приложения соответствует сайту https://futureengineer.co.uk/expertise/aerospace-engineering-recruitment-agency/
      2. WebSocket-сервер
         1. Хранит имена пользователей для всех ws-подключений;
         2. Позволяет устанавливать, закрывать ws-соединения, получать сообщения от клиентов и широковещательно рассылать сообщения подключенным клиентам;
      3. Реализация HTTP-метода Receive для получения сообщения с транспортного уровня:
         1. В json каждого сообщения указывается отправитель, время отправки, признак ошибки и полезная нагрузка;
         2. Полученное по HTTP сообщение отправляется широковещательной WebSocket рассылкой всем подключенным ws-клиентам, кроме тех, у кого логин совпадает с именем отправителя;
   2. Транспортный уровень:
      1. Реализация HTTP-метода Send для сегментирования сообщений:
         1. Разбиение сообщения на сегменты по 100 байт и их поочередная отправка на канальный уровень через метод Code;
         2. Каждый сегмент содержит время отправки (в качестве идентификатора сообщения), общую длину сообщения, номер данного сегмента в сообщении, полезную нагрузку;
      2. Реализация HTTP-метода Transfer для передачи сообщения на прикладной уровень:
         1. Формирование очереди для полученных сегментов, которые раз в 2 секунды собираются в сообщения прикладного уровня;
         2. Если часть из сегментов сообщения не была принята, оно передается на прикладной уровень с признаком ошибки;
   3. Канальный уровень:
      1. Сервис канального уровня эмулирует канал связи с потерями:
         1. Сервис должен вносить ошибку с вероятностью 10% в один случайный бит каждого сформированного кадра;
         2. Сервис должен терять передаваемый кадр с вероятностью 2%;
      2. Реализация HTTP-метода Code для кодирования и декодирования полученного от транспортного уровня сегмента:
         1. Полученный от транспортного уровня json сегмента кодируется циклическим [7,4]-кодом для получения кадра;
         2. После внесения ошибки в кадр он декодируется с исправлением ошибки и передается далее в виде сегмента на транспортный уровень;
2. **Требования к составу технических средств:**
   1. Прикладной уровень:
      1. Серверная часть
         1. ПК с ОС Windows (7.0 и выше)
         2. React.js (18.2 и выше)
         3. Material UI (5.15 и выше)
         4. Node.js (1.20 и выше)
      2. Клиентская часть
         1. ПК с ОС Windows (7.0 и выше)
         2. Веб-браузер: Chrome (40 и выше)
   2. Транспортный уровень:
      * 1. ПК с ОС Windows (7.0 и выше)
        2. Kafka (2.12 и выше)
        3. Docker (23 и выше)
        4. Go (1.24.0 и выше)
   3. Канальный уровень:
      * 1. ПК с ОС Windows (7.0 и выше)
        2. Node.js (1.20 и выше)
3. **Этапы разработки:**
   1. Выбрать тему-вариант, определить команду и разработать ТЗ – 3 неделя;
   2. Разработать макет figma (https://www.figma.com/design/msrXSyWjRCelRIu5V1Tj2S/Untitled?node-id=0-1&p=f&t=H0b9QUhhfQHJMEXQ-0), три диаграммы последовательности и описать HTTP-методы в swagger – 8 неделя;
   3. Разработать полный комплект документов – 12 неделя;
   4. Разработать и отладить приложение – 15 неделя.
4. **Техническая документация, предъявляемая по окончании работы:**
   1. Технический проект:

Расчётно-пояснительная записка, включающая в приложении комплект технической документации на программный продукт, содержащий:

* Приложение А – Техническое задание
* Приложение Б – Программа и методика испытаний
* Приложение В – Руководство пользователя
* Приложение Г – Руководство системного администратора
  1. Папка с технической и программной документацией в формате:

<группа>\_ <Фамилия И.О. студента> \_КР\_СТ\_в\_АСОИУ.zip.

1. **Порядок приёма работы:**

Прием работы осуществляется в соответствии с "Программой и методикой испытаний".

1. **Дополнительные условия:**

Данное Техническое Задание может дополняться и изменяться в установленном порядке.

# ПРИЛОЖЕНИЕ Б ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Утверждаю

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_2025г.

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Дисциплина «Сетевые технологии в АСОИУ»

Программа и методика испытаний

Вариант 25

Студенты группы ИУ5-63Б:

Буйдина К. А.

Смирнов А. А.

Хабленко И. Д.

2025г.

1. **Объект испытаний:**

Объектом испытания является распределенная информационная система обмена сообщениями в реальном времени.

1. **Цель испытаний:**

Целью проведения испытаний является доказательство работоспособности описанного в пункте 1 объекта испытаний.

1. **Требования к объекту испытаний:**

Требования к объекту испытаний представлены в документе «Техническое задание».

1. **Требования к программной документации:**

Во время проведения испытания должны быть представлены следующие документы:

1. Техническое задание;
2. Программа и методика испытаний.
3. **Программа испытаний:**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Номер пункта ТЗ | Выполняемое действие | Результат |
| 1 | 5.1.1.1 | Запуск приложения. | Приложение запущено. Открывается стартовая страница приложения. |
| 2 | 5.1.2.1 | Заполнение поля имени, нажатие на кнопку «Подключиться». | Осуществляется переход на страницу чата. |
| 3 | 5.1.2.2 | Отправка сообщений. Напишите сообщение и нажатие на кнопку «Отправить». | Доступно только пользователям с сервера Земли. Отображение имени пользователя, текста сообщения и времени отправки сообщения. Отправленные сообщения отображаются с иконкой справа. |
| 4 | 5.1.2.3 | Получение сообщения от другого пользователя. | Доступно пользователям всех серверов: Земли и Марса. Отображение имени отправителя, текст сообщения и времени отправки сообщения. Полученные сообщения отображаются с иконкой слева. |
| 5 | 5.1.2.3 | Очистка истории. | Доступно для всех серверов: Земли и Марса. При нажатии кнопки вся история отправленных сообщений удаляется. |
| 6 | 5.1.2.9 | Выход из чата. Нажатие на кнопку «Выйти». | Осуществляется переход на стартовую страницу. |
| 7 | 5.1.2.10 | Отправка сообщения с ошибкой. Нажатие на кнопку «Отправить». | Отображение значка ошибки и сообщения «При отправке сообщения возникла ошибка» |
| 8 | 5.1.3 | Запоминание имени для WebSocket подключения. Ввод имени и нажатие на кнопку «Войти». | Открытие WebSocket соединения, запоминание имени пользователя. |
| 9 | 5.1.4 | Метод Receive. | В json сообщения указывается отправитель, пэйлоад сообщения, время отправки, признак ошибки. Сообщение отправляется всем участникам чата. |
| 10 | 5.2.1 | Метод Send. Ввод сообщения, нажатие на кнопку «Отправить». | На транспортном уровне сообщение разбивается на сегменты по 100 байт и посегментно отправляется на канальный уровень. В сегменте содержатся время отправки, общая длина сообщения, номер данного сегмента, полезная нагрузка. |
| 11 | 5.2.2 | Метод Transfer. | На транспортном уровне формируется очередь из полученных сегментов. Сегменты собираются в сообщение раз в 2 секунды. Если часть сегментов не была принята, то сообщение передается на прикладной уровень с признаком ошибки. |
| 12 | 5.3.1 | Метод Code. | Полученный от транспортного уровня json сегмента кодируется Циклическим кодом [7, 4] в битовый формат. Вносится ошибка в бит сегмента. Далее сегмент декодируется, при наличии ошибки она исправляется, либо происходит потеря сегмента. Затем сегмент передается на транспортный уровень. |

# ПРИЛОЖЕНИЕ В РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Утверждаю

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_2025г.

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Дисциплина «Сетевые технологии в АСОИУ»

Руководство пользователя

Вариант 25

Студенты группы ИУ5-63Б:

Буйдина К. А.

Смирнов А. А.

Хабленко И. Д.

2025г.

* + - 1. **Введение**
  1. **Область применения**

Требования настоящего документа применяются при:

* предварительных комплексных испытаниях;
* опытной эксплуатации;
* приемочных испытаниях;
* промышленной эксплуатации.
  1. **Краткое описание возможностей**

Распределённая система обмена сообщениями, представляющая собой чат для обмена сообщения между пользователями в реальном времени.

Распределенная система состоит из двух серверов: Земля и Марс и предоставляет возможность доступа к чату всем людям, которые перешли по ссылке и ввели свое имя в строку идентификации. При успешной идентификации пользователь с любого сервера получает возможность читать сообщения, отправленные другими пользователями, начиная с того времени, как он присоединился к чату. Если пользователь с сервера Земля, то он может и сам писать и отправлять сообщения тем пользователям, которые находятся вместе с ним в чате.

Актуальность переписки поддерживается при помощи протокола WebSocket, который позволяет обновлять окно чата с сообщениями в реальном времени.

1. **Назначение и условия применения распределенной системы**
   1. **Назначение распределенной системы**

Распределенная система предназначена для возможности вести переписку нескольким пользователям в реальном времени.

Для использования Распределенной системы необходимо выполнение следующих условий:

* 1. **Системные требования**

Для работы необходим Yandex Browser или Google Chrome.

1. **Условия выполнения программы**

Для работы программы требуется браузер Google Chrome или любой иной поддерживающий современные функции JavaScript, а также стабильное интернет-соединение.

1. **Выполнение программы**
   1. **Инсталяция/деинсталяция**

Потребуется Web-браузер, рекомендуется, Yandex Browser или Google Chrome.

* 1. **Запуск программы**

На компьютере необходимо запустить браузер и ввести в адресную строку http://localhost:3000/ для подключению к серверу Земли и http://localhost:3005/ для подключению к серверу Марса.

1. **Описание операций**
   1. **Идентификация в системе**

Доступно для: все пользователи.

Операция: открыть веб страницу.

Для идентификации в системе необходимо открыть страницу и ввести свое имя, а также нажать на кнопку «Подключиться». (рис.1).

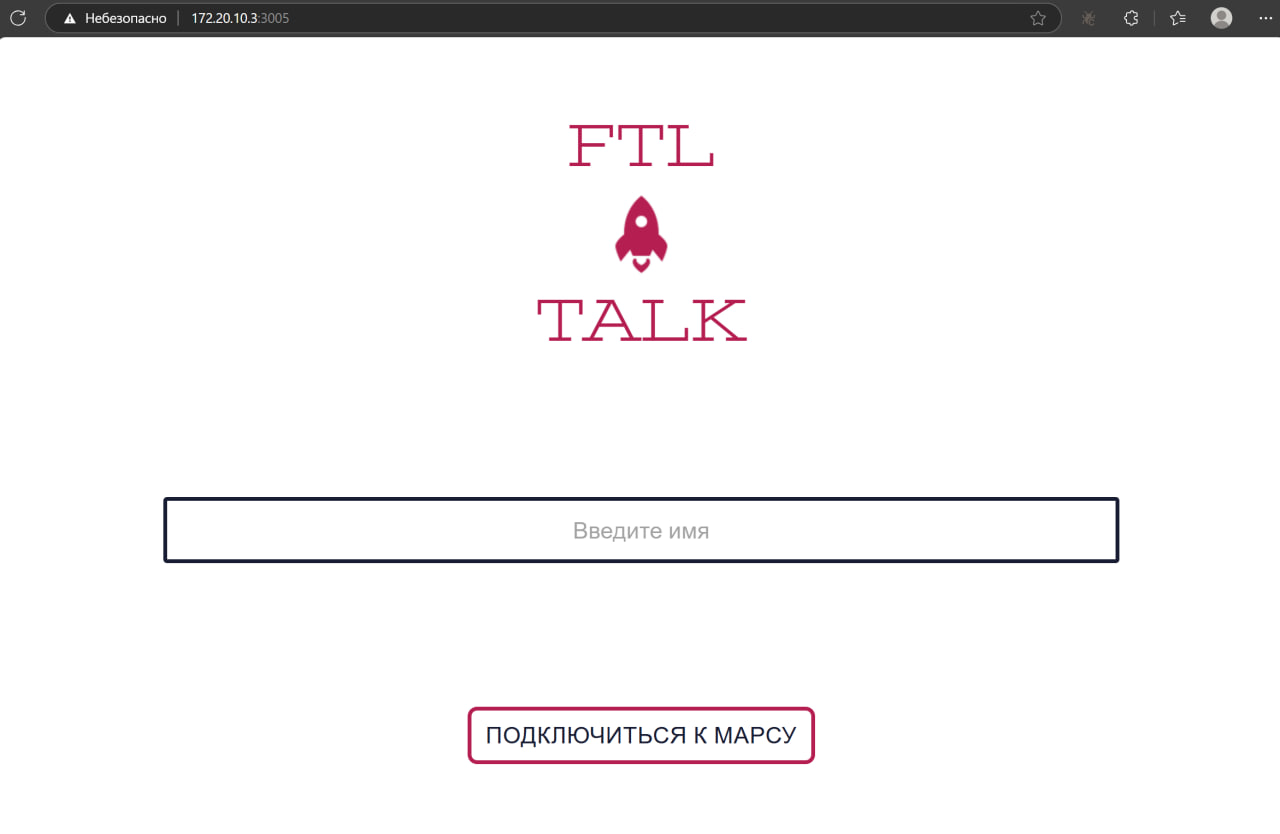


Рисунок 1 – окно идентификации

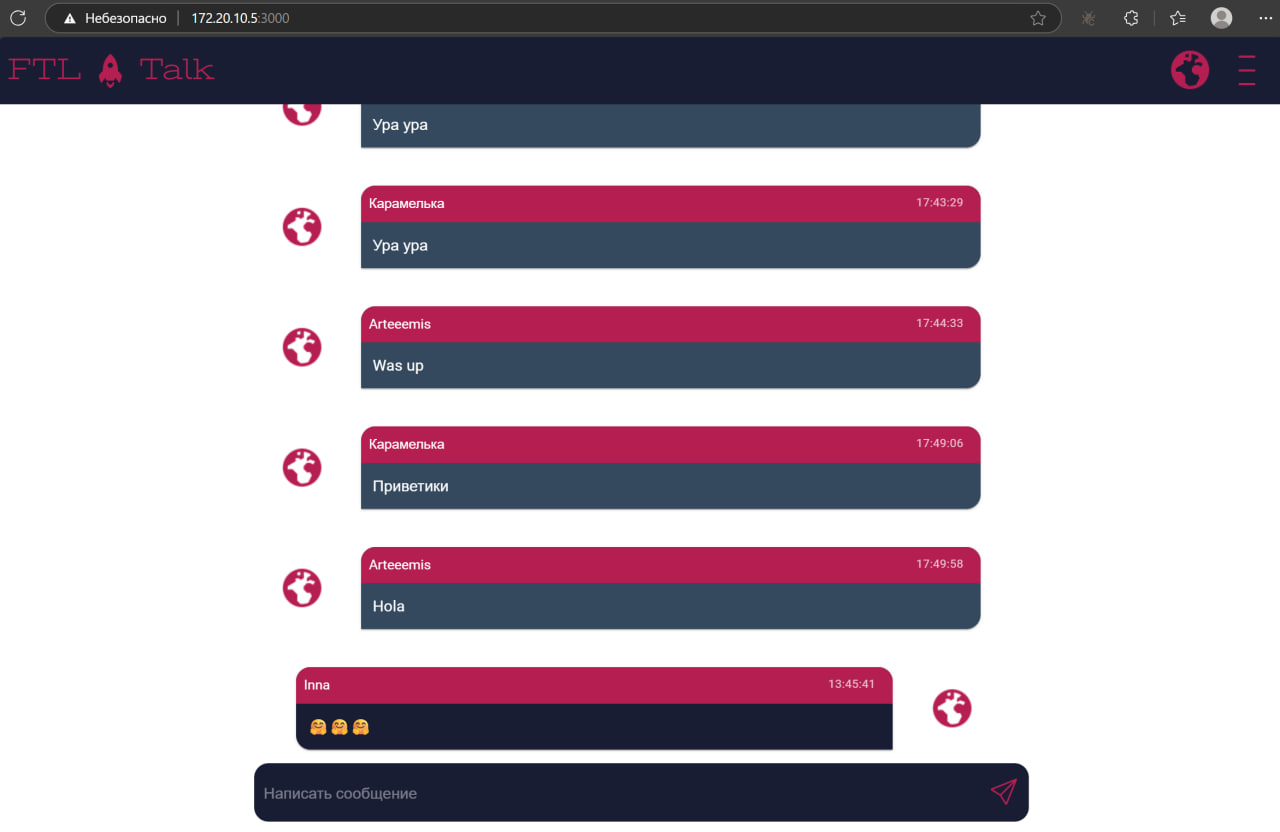


Рисунок 2 – авторизованный пользователь

* 1. **Просмотр и отправка сообщений**

Доступно для: идентифицированным пользователям.

Операция 1: для отправки сообщения нужно вести его в специальное поле для сообщений и нажать на значок самолета, чтобы отправить его (рис.2).





Рисунок 3 – отправка сообщения



Рисунок 4 – сообщение отправлено

Операция 2: для получения сообщений от других пользователей нужно находиться на странице чата и ждать сообщения. (рис.5).



Рисунок 5 – окно чата

* 1. **Выход из чата**

Доступно для: идентифицированным пользователям.

Операция: для выхода из чата необходимо нажать кнопку выхода в правом верхнем углу хэдера.

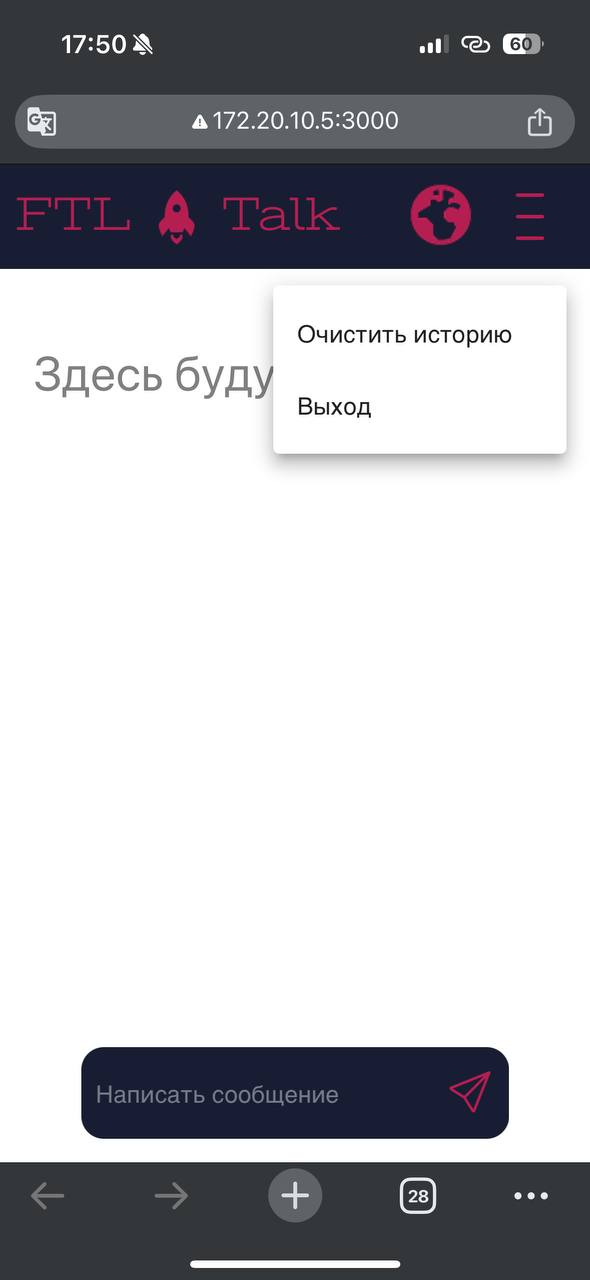


Рисунок 6 – кнопка «выйти»

# ПРИЛОЖЕНИЕ Г РУКОВОДСТВО СИСТЕМНОГО АДМИНИСТРАТОРА

|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Утверждаю

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_\_"\_\_\_\_\_\_\_\_\_2025г.

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Дисциплина «Сетевые технологии в АСОИУ»

Руководство системного администратора

Вариант 25

Студенты группы ИУ5-63Б:

Буйдина К. А.

Смирнов А. А.

Хабленко И. Д.

2025г.

1. **Системные требования:**
   1. Прикладной уровень:
      1. ПК с ОС Windows (7 и выше)
      2. React.js (18.2 и выше)
      3. Material UI (5.15 и выше)
      4. Node.js (1.20 и выше)
   2. Транспортный уровень:
      1. ПК с ОС Windows (7 и выше)
      2. Kafka (2.12 и выше)
      3. Docker (23 и выше)
      4. Go (1.24.0 и выше)
   3. Канальный уровень:
      1. ПК с ОС Windows (7 и выше)
      2. Node.js (1.20 и выше)
2. **Инструкция по развертыванию системы:**
   1. Прикладной уровень:
      1. Клонирование репозитория:  
         git clone https://github.com/Arteeemis/FTL-TALK-APPLICATION
      2. Установка зависимостей сервера Земли:  
         cd WebSocketServer  
         npm install
      3. Установка зависимостей сервера Марса

cd WebSocketServerMars

npm install

* + 1. Запуск серверов:  
       ts-node index.ts
    2. Установка зависимостей клиента Земли:  
       cd frontend  
       npm install
    3. Установка зависимостей клиента Марса:

cd frontend-mars

npm install

* + 1. Запуск пользовательских приложений:  
       npm run start
  1. Транспортный уровень:
     1. Клонирование репозитория:  
        git clone https://github.com/KristinaBu/mars-chat-transport.git
     2. Развёртывание контейнера:  
        docker compose up
     3. Установка зависимостей:  
        go mod tidy
     4. Запуск проекта  
        go run main.go
  2. Канальный уровень:
     1. Клонирование репозитория:   
        git clone https://github.com/Kh-Inna/mars-chat-channel.git
     2. Установка зависимостей:

npm install

* + 1. Запуск проекта

node app.js