|  |  |
| --- | --- |
|  | **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 \_\_ИУ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_Б.В. Ларкин\_\_\_**

(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 \_\_ИУ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_С.С. Серов\_\_\_**

(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 \_\_ИУ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_ Т.А. Цыпышев \_\_\_**

(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\_\_\_\_\_

(подпись) (И.О.Фамилия)

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

**ЗАДАНИЕ**

**на выполнение курсовой работы**

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

Студент группы \_ИУ5-61Б\_\_\_Серов С.С., Цыпышев Т.А., Ларкин Б.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(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 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ С.С. Серов \_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_Т.А. Цыпышев\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_ Б.В. Ларкин\_\_\_\_\_\_

(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 | ИУ5-61Б Серов С.С., Цыпышев Т.А., Ларкин Б.В. | |
|  | (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 | *\_25.02.24\_*  *Планируемая дата* | *21.02.24* |  |  |
| 2. | Создание макета Figma, диаграммы последовательности и формирование swagger | *31.03.24\_*  *Планируемая дата* | *29.03.24* |  |  |
| 3. | Оформление РПЗ, ПМИ, РСА, РП | *28.04.24\_*  *Планируемая дата* | *27.04.24* |  |  |
| 4. | Тестирование распределенной системы и подготовка проекта к демонстрации | *15.05.24\_*  *Планируемая дата* |  |  |  |
| 5. | Защита курсовой работы | *16.05.24\_*  *Планируемая дата* |  |  |  |

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

(подпись, дата) (подпись, дата)

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

(подпись, дата)

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

(подпись, дата)

# **ВВЕДЕНИЕ**

В современную эпоху стремительного технологического прогресса компании и частные пользователи всё чаще сталкиваются с задачей оперативного обмена данными. К 2025 году значительное число организаций, специализирующихся на создании цифровых решений, запускают собственные мессенджеры для обсуждения профессиональных вопросов. Подобная тенденция обусловлена как быстрым развитием IT-сферы и расширением глобальных связей, так и возросшими требованиями к защите информации в чатах.

В связи с этим возрастает необходимость в разработке надежных платформ для обмена сообщениями и файлами, которые бы сочетали в себе высокую скорость передачи данных и высокий уровень безопасности. Решением задачи может стать построение трёхуровневой системы, включающей прикладной, транспортный и канальный уровни. Прикладной уровень будет отвечать за удобство взаимодействия пользователей благодаря современному интерфейсу. Транспортный займется стабильной доставкой информации, а канальный уровень проследит за её защитой и целостностью.

В итоге создание такой системы сможет в полной мере удовлетворить требования к обмену информацией в режиме реального времени, что становится всё более актуальным.

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 отвечает за передачу данных и функционирует как механизм обмена сообщениями. Помимо этого, он играет роль посредника между прикладным и канальным уровнями. Диаграммы последовательности и развертывания, представленные на рисунках 1 и 2 соответственно, наглядно демонстрируют его работу. Из них видно, что транспортный уровень должен справляться с высокой нагрузкой, обеспечивая бесперебойное взаимодействие между двумя бэкендами и предотвращая возникновение «узких мест» в системе.

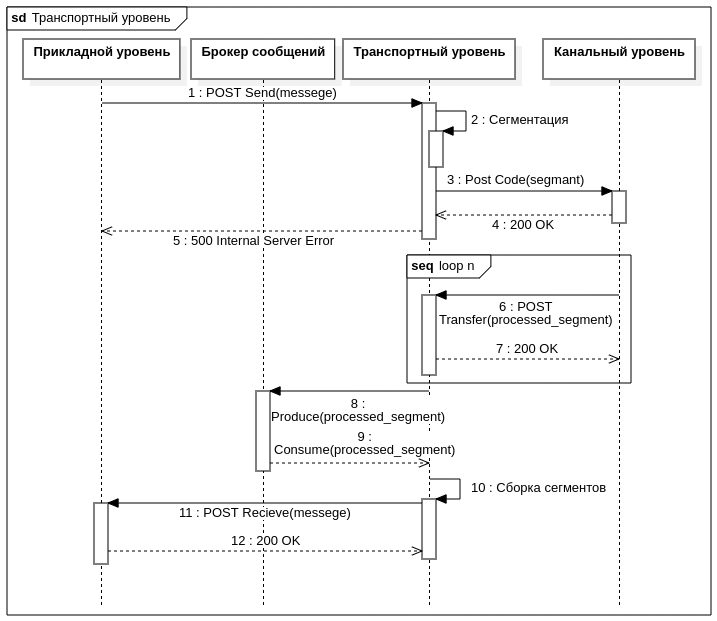


Рисунок 1 – Диаграмма последовательности транспортного уровня

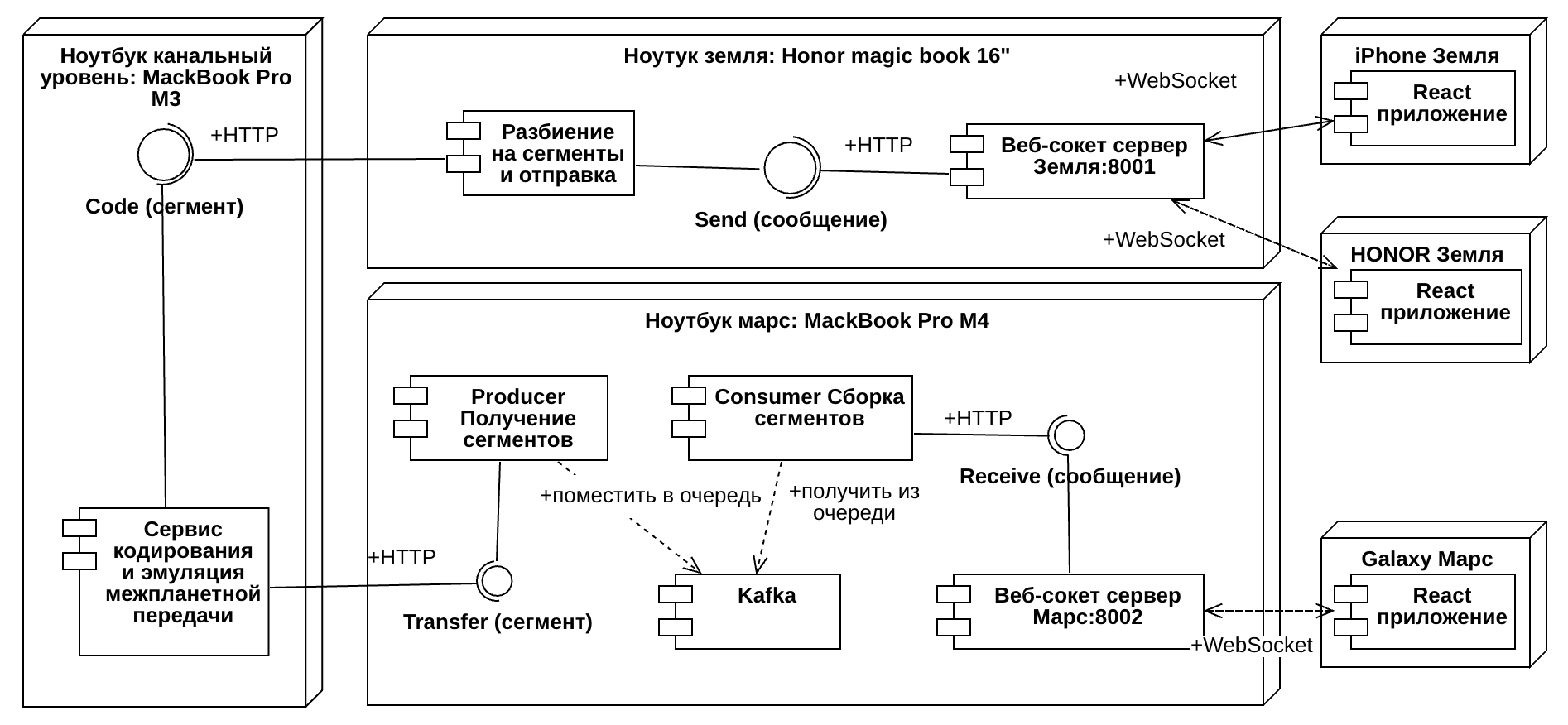


Рисунок 2 – Диаграмма развёртывания

Рассмотрим более детально принцип функционирования транспортного уровня. Процесс начинается с того, что прикладной уровень инициирует вызов метода Send, передавая в запросе имя отправителя, изображение в закодированном строковом формате и метку времени, которая служит идентификатором сообщения. После этого данные разбиваются на сегменты объёмом до 140 байт. Каждый сегмент включает в себя полезную нагрузку, временную метку, порядковый номер сегмента и общее количество сегментов. Далее сегменты поочередно обрабатываются методом Code на канальном уровне. После кодирования канальный уровень возвращает обработанные сегменты через вызов метода Transfer. При возникновении ошибок в процессе обработки в запрос может быть добавлен специальный флаг, сигнализирующий пользователю о том, что сообщение повреждено и его невозможно корректно прочитать.

Одним из ключевых аспектов транспортного уровня в модели OSI является механизм управления перегрузкой (congestion control). Он включает в себя набор алгоритмов, оптимизирующих скорость передачи данных через протокол TCP между двумя узлами сети [2]. Эти алгоритмы контролируют размер TCP-окна и ориентируются на такие параметры, как Round Trip Time (RTT), потери пакетов и задержки отправки данных [7]. В рамках рассматриваемой упрощённой реализации транспортного уровня используется очередь отправки пакетов: полученные от канального уровня сегменты помещаются в систему Kafka. Kafka представляет собой распределённую платформу обмена сообщениями с высокой отказоустойчивостью, основанную на модели «публикация/подписка» для обработки потоков событий. Далее каждые две секунды сегменты объединяются в полное сообщение и отправляются обратно на прикладной уровень методом Receive. В случаях, когда в течение двух циклов не удаётся получить все необходимые сегменты, сообщение всё равно передаётся с отметкой об ошибке.

Методы, реализуемые транспортным уровнем, представлены в таблице 1:

**Таблица 1 – Методы транспортного уровня**

|  |  |  |
| --- | --- | --- |
| Метод | Входные параметры | Описание |
| POST Send | username string  data string  send\_time integer | Отправка сообщения с прикладного уровня на транспортный |
| POST Transfer | segment\_number integer  total\_segments integer  username string  send\_time string  payload string | Отправка декодированного сегмента с канального уровня на транспортный |

# **КАНАЛЬНЫЙ УРОВЕНЬ**

На этом уровне моделируется взаимодействие с удалённым сетевым узлом через канал, подверженный шумам и помехам. Для защиты передаваемых данных применяется код Хэмминга [7,4]. Полученный от транспортного уровня JSON-сегмент преобразуется в битовый формат, затем подвергается кодированию с использованием указанного кода, после чего в него искусственно вносятся ошибки. Далее выполняется декодирование для последующей передачи обратно на транспортный уровень.

Код Хэмминга представляет собой механизм самопроверки и самовосстановления, разработанный для двоичной системы счисления. Он позволяет исправить одиночные ошибки и выявить случаи двойных ошибок. Иными словами, данный алгоритм кодирует сообщение таким образом, что после его передачи можно определить наличие ошибок (например, вызванных помехами) и, в случае возможности, восстановить исходные данные [6]. В процессе кодирования в исходное сообщение вставляются дополнительные контрольные биты — обычно на позиции, соответствующие степеням двойки. Этот метод эффективен в средах с преимущественно одноразрядными ошибками и отличается простотой реализации, что делает его особенно удобным для использования в компьютерной памяти и сетях передачи данных. Основным недостатком кода Хэмминга является риск некорректного исправления при наличии многобитных ошибок, что может привести к ещё большим искажением данных. Кроме того, для корректной работы требуется добавление избыточных битов. Например, для восьмибитного сообщения необходимо добавить четыре контрольных бита, что увеличивает его объём на 50%, а для шестнадцати бит — пять бит.

На канальном уровне обрабатываются сегменты данных объёмом 140 байт. Вероятность внесения ошибок в передаваемые данные составляет 10%, а вероятность потери кадра — 2%.

Для реализации данного уровня был выбран язык программирования Go [5]. В рамках решения задачи был разработан сервер, работающий на порту 5000 с эндпоинтом «code», а также создан отдельный пакет для работы с битовыми данными, их кодирования и последующего декодирования.

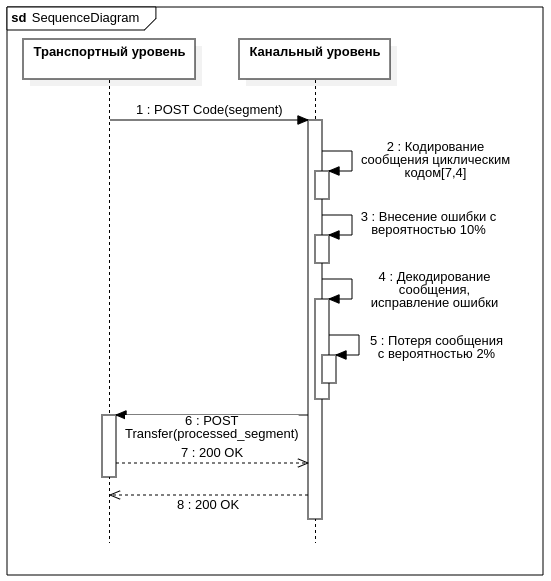


Рисунок 3 – Диаграмма последовательности канального уровня

Метод, используемый на канальном уровне (Таблица 2):

**Таблица 2 – Методы канального уровня**

| Метод | Входные параметры | Описание |
| --- | --- | --- |
| POST Code | segment\_number integer  total\_segments integer  username string  send\_time string  payload string | Полученный от транспортного уровня json сегмента кодируется кодом Хэмминга [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. При подключении вводится имя пользователя и выполняется установка WebSocket соединения. Чат является общим и не хранит историю сообщений. Если сообщение доставляется с признаком ошибки, оно игнорируется.

Для реализации пользовательского интерфейса прикладного уровня используется React [3, 4] с менеджером состояний Redux Toolkit и с библиотекой для запросов Axios. В качестве UIkit взят MUI, а за основу дизайна выбран Stepik. Для сервера используется Node.js с фрейворком Express.js [1].

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.

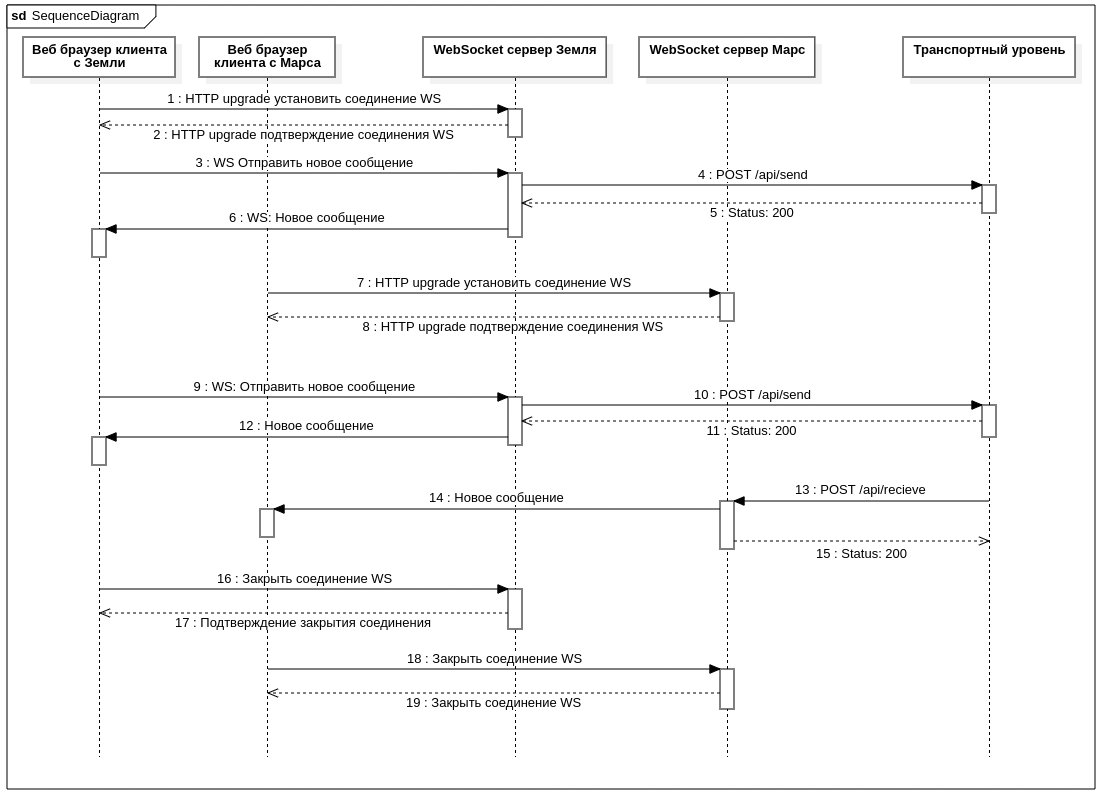
В нашем случае WS-сервера два: один работает для пользователей Марса, другой – для пользователей с Земли. В то время как земляне могут общаться не выходя за рамки прикладного уровня, для пересылки сообщений марсианам понадобится вызов запросов написанного API.

Рисунок 4 – Диаграмма последовательности прикладного уровня

**Таблица 3 – Методы прикладного уровня**

|  |  |  |
| --- | --- | --- |
| Метод | Входные параметры | Описание |
| POST Receive | send\_time string  payload string  sender string  error boolean  errormsg: string | Прием декодированного сегмента (сообщения) с транспортного уровня на прикладной |

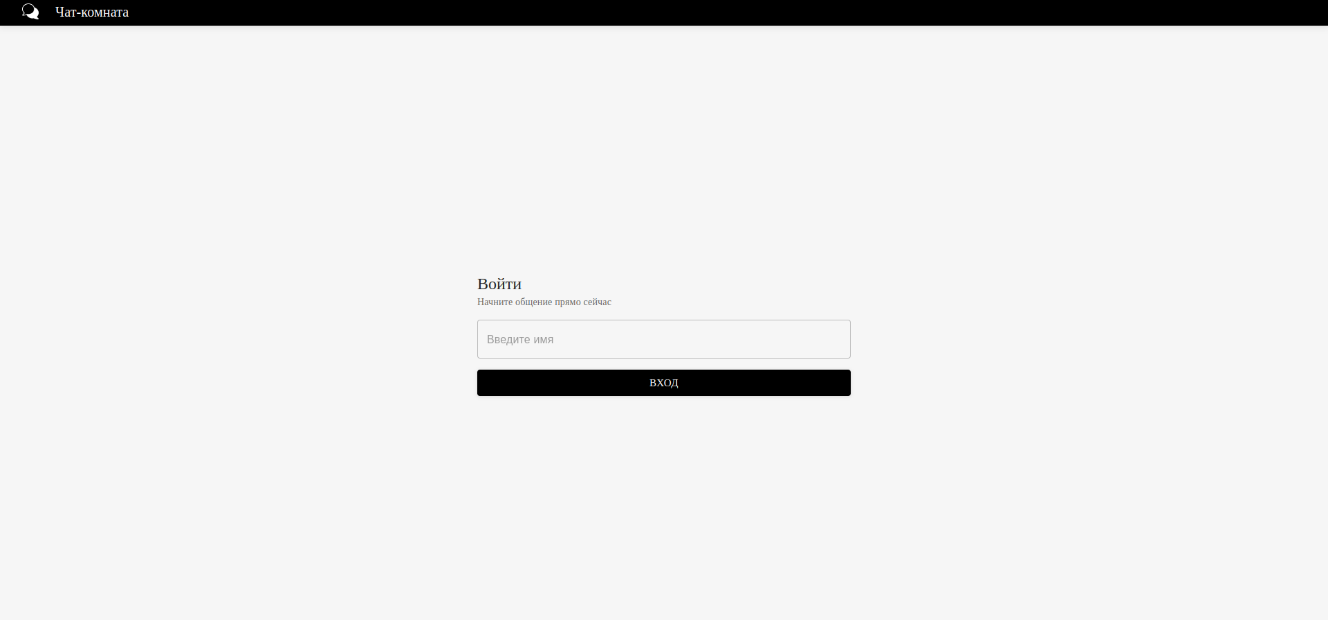


Рисунок 5 – Скриншот работы программы (авторизация)

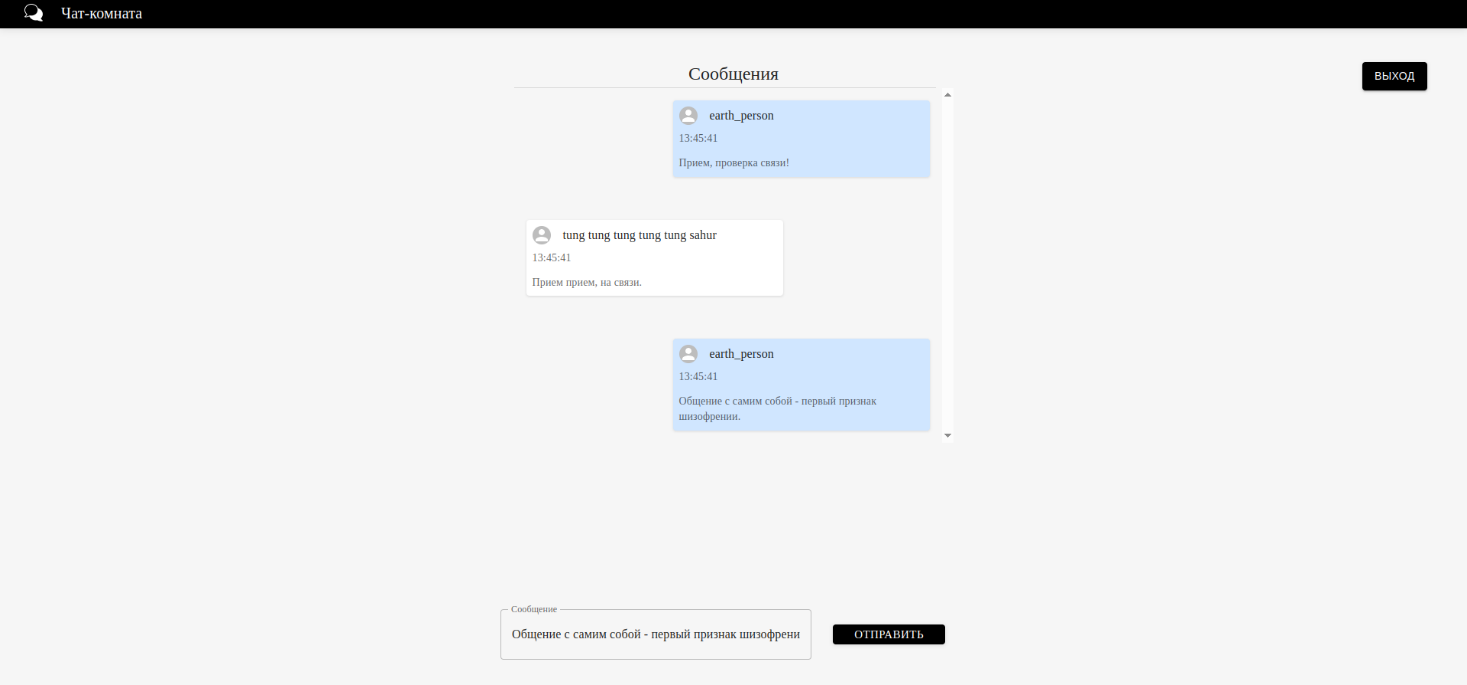


Рисунок 6 – Скриншот работы программы

**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. JavaScript полное руководство [Книга] // Д.Флэнаган. (дата обращения 05.03.2025).
  2. Полное практическое руководство по Docker [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/310460 (дата обращения: 17.03.2025).
  3. Руководство по React [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/ruvds/articles/428077 (дата обращения: 05.03.2025).
  4. Quick Start – React [Электронный ресурс] // React. URL: https://react.dev/learn (дата обращения: 05.03.2025).
  5. An Introduction to Programming in Go [Электронный ресурс] // Go Resources. URL: https://www.golang-book.com/books/intro (дата обращения: 13.03.2025).
  6. Код Хэмминга. Пример работы алгоритма [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/140611/ (дата обращения: 17.03.2025).
  7. TCP Congestion Control или почему скорость прыгает [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/168407/ (дата обращения: 03.04.2025).
  8. Apache Kafka: обзор [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/piter/articles/352978/ (дата обращения: 03.04.2025).

**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 33

Студенты группы ИУ5-61Б:

Цыпышев Т.А.

Серов С.С.

Ларкин Б.В.

2025г.

1. **Наименование:**Разработать распределенную систему обмена файлами в реальном времени.
2. **Основание для разработки:**Основанием для разработки является учебный план МГТУ им. Баумана кафедры ИУ5 на 6 семестр.
3. **Исполнители:**Ларкин Б.В. (прикладной уровень) – группа ИУ5-61Б  
   Цыпышев Т.А. (транспортный уровень) – группа ИУ5-61Б  
   Серов С.С. (канальный уровень) – группа ИУ5-61Б
4. **Цель разработки:**Разработать распределенную систему для обмена сообщениями в реальном времени, состоящую из трех уровней: прикладной, транспортный и канальный, каждый из которых реализуется как отдельный веб-сервис.
5. **Функциональные требования**
   1. Прикладной уровень:
      1. Страница приложения представляет собой чат, где пользователи могут отправлять и получать текстовые сообщения. Каждое сообщение будет отображать информацию об отправителе и времени отправки.
         1. Соединение с сервером Земли или Марса выбирается посредством подключения к порту 8001 или 8002 веб-приложения.
         2. При подключении к чату пользователю следует ввести свое имя. Это имя будет передаваться вместе с каждым новым текстовым сообщением;
         3. После успешной авторизации пользователи с "Земли" смогут отправлять текстовые сообщения на Марс через установленное WebSocket-соединение и напрямую на Землю. Для ввода текста предусмотрено поле ввода. Для отправки сообщения предусмотрена кнопка "Отправить";
         4. У пользователей с Марса отсутствует возможность отправки сообщений. При попытке отправки пользователь увидит ошибку от сервера.
         5. Если при получении сообщения возникнет ошибка, то это сообщение не будет отображаться, а пользователи увидят вместо него уведомление о сбое;
         6. При нажатии кнопки "Выйти" все данные (чат и логин) очищаются, а активное WebSocket-соединение закрывается;
         7. Дизайн приложения должен соответствовать сайту stepik.org и быть адаптирован для мобильных устройств: <https://www.figma.com/design/ecp3I8tS1ORT2gIRPeXJ93/Material-UI-for-Figma-(and-MUI-X)-(Community)?node-id=8603-8199&t=10V4Q609mCz5jgWP-1>
      2. WebSocket-сервера
         1. WebSocket-сервера управляют списком пользователей, поддерживающих активные соединения и принадлежащих данному серверу. Они хранят информацию об именах пользователей для каждого активного соединения;
         2. Сервера предоставляют возможность открывать и закрывать WebSocket-соединения. Полученные сообщения от клиентов с Земли сервер рассылает всем подключенным пользователям с Земли, за исключением тех, чье имя совпадает с именем отправителя, а также через многоуровневую структуру приложения — клиентам на Марсе;
      3. Реализация HTTP-метода Receive для получения сообщения с транспортного уровня:
         1. Каждое сообщение, передаваемое по HTTP, содержит информацию о том, кто является отправителем, когда было отправлено сообщение, есть ли ошибка в передаче, а также сам текст сообщения;
         2. После получения сообщения по HTTP, сервер передает его всем подключенным WebSocket-клиентам через широковещательную рассылку, кроме того клиента, который отправил это сообщение или клиентов, которые находятся на Земле;
   2. Транспортный уровень:
      1. Реализация HTTP-метода Send, который отвечает за разделение текстового сообщения на несколько частей (сегментов):
         1. Каждое сообщение будет разделяться на сегменты по 140 байт. Каждый сегмент будет последовательно передаваться через метод Code на канальный уровень;
         2. Каждый сегмент будет содержать информацию о времени отправки сообщения (чтобы все части можно было собрать в правильном порядке), общую длину сообщения, порядковый номер сегмента и сам текст сообщения;
      2. Реализация HTTP-метода Transfer, который собирает полученные сегменты в полное сообщение прикладного уровня:
         1. Полученные сегменты будут храниться в очереди, и раз в 1 секунду сервер будет собирать их в одно сообщение и передавать на прикладной уровень;
         2. Если некоторые сегменты не были получены или переданы с ошибкой, сервер будет отправлять полное сообщение с признаком ошибки;
   3. Канальный уровень:
      1. Сервис канала связи имитирует возможные потери данных. Он добавляет случайные ошибки в передаваемые сообщения:
         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, который выполняет кодирование и декодирование полученных от транспортного уровня сегментов:
         1. Каждый сегмент, полученный от транспортного уровня, кодируется с использованием циклического [7,4]-кода, что позволяет обнаруживать и исправлять ошибки при передаче данных;
         2. После того как ошибка внесена в кадр, он будет декодирован, исправлен и передан дальше на транспортный уровень в виде конкретного сегмента;
6. **Требования к составу технических средств:**
   1. Прикладной уровень:
      1. Серверная часть
         1. ПК с ОС Linux(5.2 и выше)
         2. Node.js (1.20 и выше)
      2. Клиентская часть
         1. Телефон с ОС Android(10 и выше)
         2. Веб-браузер: Chrome(100 и выше)
   2. Транспортный уровень:
      1. ПК с ОС MacOS(6.4 и выше)
      2. GoLang (4.2 и выше)
      3. Kafka (2.12 и выше)
   3. Канальный уровень:
      1. ПК с ОС MacOS(6.4 и выше)
      2. GoLang (4.2 и выше)
7. **Этапы разработки:**
   1. Выбрать тему-вариант, определить команду и сформировать ТЗ – 3 неделя;
   2. Разработать макет figma, три диаграммы последовательности и описать HTTP-методы в swagger – 8 неделя;
   3. Разработать и отладить приложение, подготовить полный комплект документов (ТЗ, РПЗ, ПМИ, РП, РСА) – 12 неделя;
   4. Исправить замечания, защитить проект – 14 неделя.
8. **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 приема работы:**Приемка работы осуществляется в соответствии с "Программой и методикой испытаний."
2. **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 33

Студенты группы ИУ5-61Б:

Цыпышев Т.А.

Серов С.С.

Ларкин Б.В.

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 «Войти». | Осуществляется переход на страницу чата. |
| 3 | 5.1.2.2 | Отправка сообщений.  Ввод текста сообщения и нажатие на кнопку «Отправить». | Отображение имени пользователя и текста сообщения, времени отправки; голубым цветом отображение фона сообщения. |
| 4 | 5.1.2.3 | Получение сообщения от другого пользователя. | Отображение имени отправителя и текста сообщения, времени отправки; серым цветом отображение фона сообщения. |
| 5 | 5.1.2.4 | Выход из чата. Нажатие на кнопку «Выйти». | Осуществляется переход на стартовую страницу. |
| 6 | 5.1.2.10 | Отправка сообщения с ошибкой. Нажатие на кнопку «Отправить». | Отображение значка ошибки и сообщения «При отправке сообщения возникла ошибка» |
| 7 | 5.1.3 | Запоминание имени для WebSocket подключения. Ввод имени и нажатие на кнопку «Войти». | Открытие WebSocket соединения, запоминание имени пользователя. |
| 8 | 5.1.4 | Метод Receive. | В json сообщения указывается отправитель, сообщение, id, время отправки, признак ошибки. Сообщение отправляется всем участникам чата. |
| 9 | 5.2.1 | Метод Send. Ввод сообщения, нажатие на кнопку «Отправить». | На транспортном уровне сообщение разбивается на сегменты по 140 байт и посегментно отправляется на канальный уровень. В сегменте содержатся время отправки, общая длина сообщения, номер данного сегмента, полезная нагрузка. |
| 10 | 5.2.2 | Метод Transfer. | На транспортном уровне формируется очередь из полученных сегментов. Сегменты собираются в сообщение раз в 1 секунду. Если часть сегментов не была принята, то сообщение передается на прикладной уровень с признаком ошибки. |
| 11 | 5.3.1 | Метод Code. | Полученный от транспортного уровня json сегмента кодируется кодом Хэмминга [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 33

Студенты группы ИУ5-61Б:

Цыпышев Т.А.

Серов С.С.

Ларкин Б.В.

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 предоставляет возможность доступа к чату на Марсе или на Земле всем людям, которые перешли по ссылке и ввели свое имя в строку идентификации. При успешной идентификации пользователь получает возможность читать сообщения, отправленные другими пользователями, начиная с того времени, как он присоединился к чату. Писать сообщения для отправки в земной чат и на марс могут только пользователи с земли.

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 на Земле и ретрансляции этих сообщений на Марс, все 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. **Инсталляция/деинсталляция**

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

* 1. **Запуск программы**

На компьютере необходимо запустить браузер и ввести в адресную строку URL. Для Земли: <http://localhost:8081/> , для Марса: <http://localhost:8082/> .

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 (рис.1).

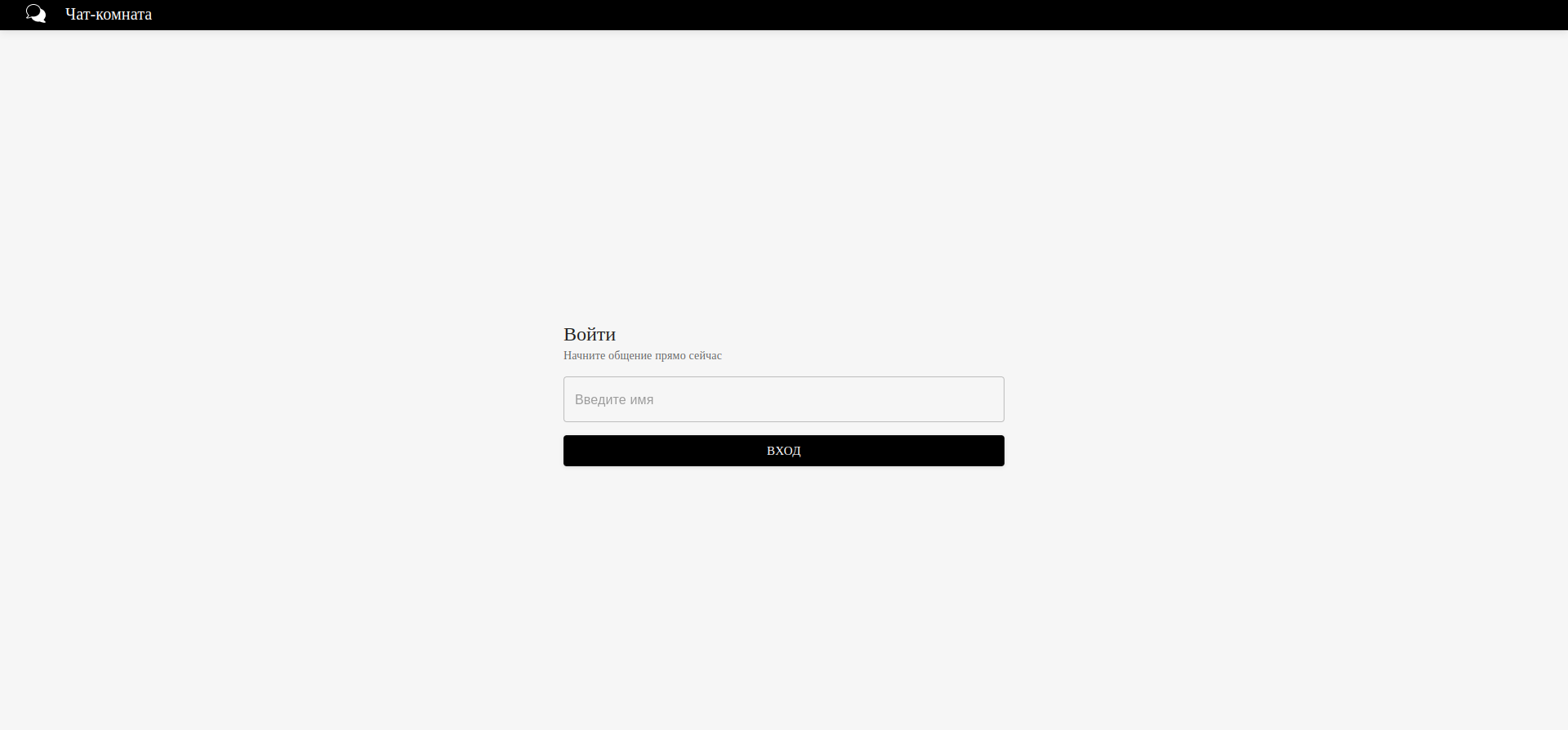


Рисунок 1 – окно идентификации

* 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 «Отправить» (рис. 2).

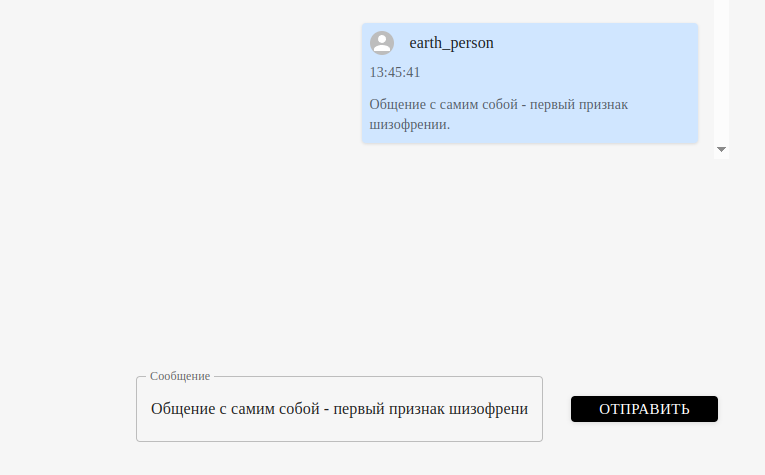


Рисунок 2 – отправка сообщения

Операция 2: для получения сообщений от других пользователей нужно находиться на странице чата и ждать сообщения. (рис. 3).

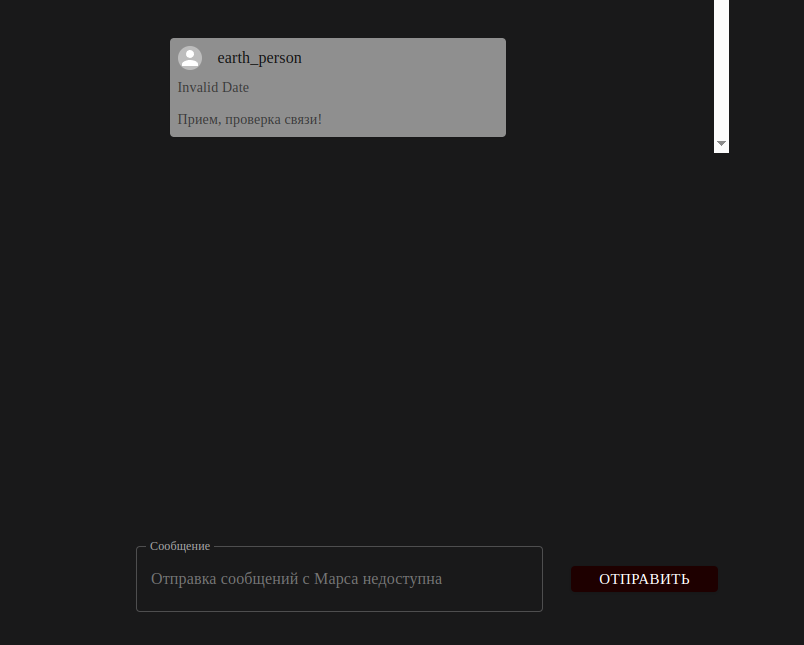


Рисунок 3 – окно чата на Марсе

* 1. **Выход из чата**

Доступно для: идентифицированным пользователям.

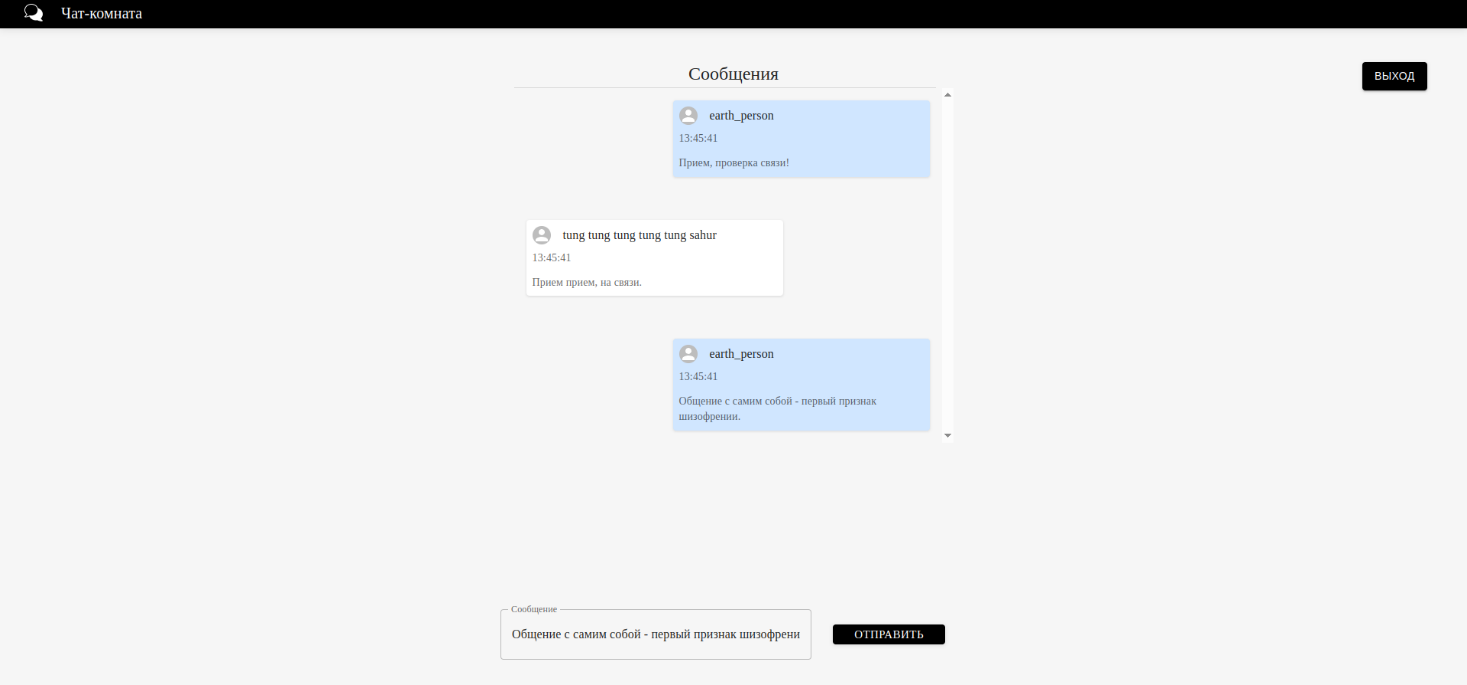
Операция: для выхода из чата необходимо нажать кнопку «выйти» (рис. 4) в правом верхнем углу страницы чата.

Рисунок 4 – 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 33

Студенты группы ИУ5-61Б:

Цыпышев Т.А.

Серов С.С.

Ларкин Б.В.

2025г.

1. **Системные требования:**
   1. Прикладной уровень:
      1. ПК с ОС Linux(5.2 и выше)
      2. Node.js (1.20 и выше)
   2. Транспортный уровень:
      1. ПК с ОС MacOS(6.4 и выше)
      2. GoLang (4.2 и выше)
      3. Kafka (2.12 и выше)
   3. Канальный уровень:
      1. ПК с ОС MacOS(6.4 и выше)
      2. GoLang (4.2 и выше)
2. **Инструкция по развертыванию системы:**
   1. Прикладной уровень:
      1. Клонирование репозитория:  
         git clone https://github.com/bmstu-enterprise-chat/application-layer
      2. Установка зависимостей сервера:  
         cd server  
         npm install
      3. Запуск сервера:  
         ts node index.ts
      4. Установка зависимостей клиента:  
         cd ../frontend  
         npm install
      5. Запуск пользовательского приложения:  
         npm run start
   2. Транспортный уровень:
      1. Клонирование репозитория:  
         git clone https://github.com/bmstu-enterprise-chat/transport-layer  
         cd network\_transport\_level
      2. Развёртывание контейнера:  
         docker compose up
      3. Установка зависимостей:  
         go mod download
      4. Сборка проекта:  
         go build -o app ./...
      5. Запуск проекта  
         ./app
   3. Канальный уровень:
      1. Клонирование репозитория:   
         git clone https://github.com/bmstu-enterprise-chat/channel-level
      2. Установка зависимостей:  
         go mod download
      3. Сборка проекта:  
         go build -o app ./...
      4. Запуск проекта  
         ./app