|  |  |
| --- | --- |
| **­­** | **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 \_\_\_\_\_\_\_\_\_\_\_РАДИОТЕХНИЧЕСКИЙ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

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

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

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

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

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

Студент \_\_РТ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_Р.М. Толкачев\_\_\_**

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

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

(Подпись, дата) (И.О.Фамилия)

*2024 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 \_\_\_\_\_\_\_\_\_\_\_РАДИОТЕХНИЧЕСКИЙ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

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

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

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

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

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

Студент \_\_РТ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_И.С. Викулин\_\_\_**

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

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

(Подпись, дата) (И.О.Фамилия)

*2024 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 \_\_\_\_\_\_\_\_\_\_\_РАДИОТЕХНИЧЕСКИЙ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

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

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

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

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

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

Студент \_\_РТ5-61Б\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_А.К. Билалов\_\_\_**

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

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

(Подпись, дата) (И.О.Фамилия)

*2024 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\_\_\_\_\_\_\_ 2024 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

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

Студент группы \_РТ5-61Б\_\_\_Викулин И.С, Билалов А.К., Толкачев Р.М.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(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\_\_\_\_\_\_\_ 2024 г.

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

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ И.С. Викулин \_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ А.К. Билалов\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_**\_\_\_ Р.М. Толкачев\_\_\_

(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 | РТ5-61Б Викулин И.С, Билалов А.К., Толкачев Р.М. | |
|  | (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* | *19.02.24* |  |  |
| 2. | Создание макета Figma, диаграммы последовательности и формирование swagger | *31.03.24\_*  *Планируемая дата* | *31.03.24* |  |  |
| 3. | Оформление РПЗ, ПМИ, РСА, РП | *28.04.24\_*  *Планируемая дата* | *29.04.24* |  |  |
| 4. | Тестирование распределенной системы и подготовка проекта к демонстрации проектапрезеннеобходимости) | *15.05.24\_*  *Планируемая дата* | *15.05.24* |  |  |
| 5. | Защита курсовой работы | *16.05.24\_*  *Планируемая дата* | *16.05.24* |  |  |

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

(подпись, дата) (подпись, дата)

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

(подпись, дата)

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

(подпись, дата)

# **ВВЕДЕНИЕ**

В XXI веке все больше компаний и частных лиц сталкиваются с необходимостью обмениваться информацией в режиме реального времени. В 2024 году большая часть компаний, занимающихся разработкой цифровых продуктов создают свои мессенджеры для обсуждения рабочих вопросов. Рост этой потребности обусловлен стремительным развитием информационных технологий и всё более глобальным характером бизнеса, а также вопросом конфиденциальности чатов.

В этом контексте возникает потребность в создании эффективной системы обмена сообщениями и файлами, которая бы обеспечивала как быстрое, так и безопасное общение. Разработка комплексной системы, состоящей из трех уровней - прикладного, транспортного и канального, может решить эти задачи. На прикладном уровне пользователи будут иметь возможность удобно общаться через современный и интуитивно понятный интерфейс. Транспортный уровень обеспечит надежную передачу данных, а канальный уровень гарантировать их целостность и безопасность.

Таким образом, разработка такой системы позволит эффективно удовлетворить растущие потребности в обмене информацией в реальном времени.

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».

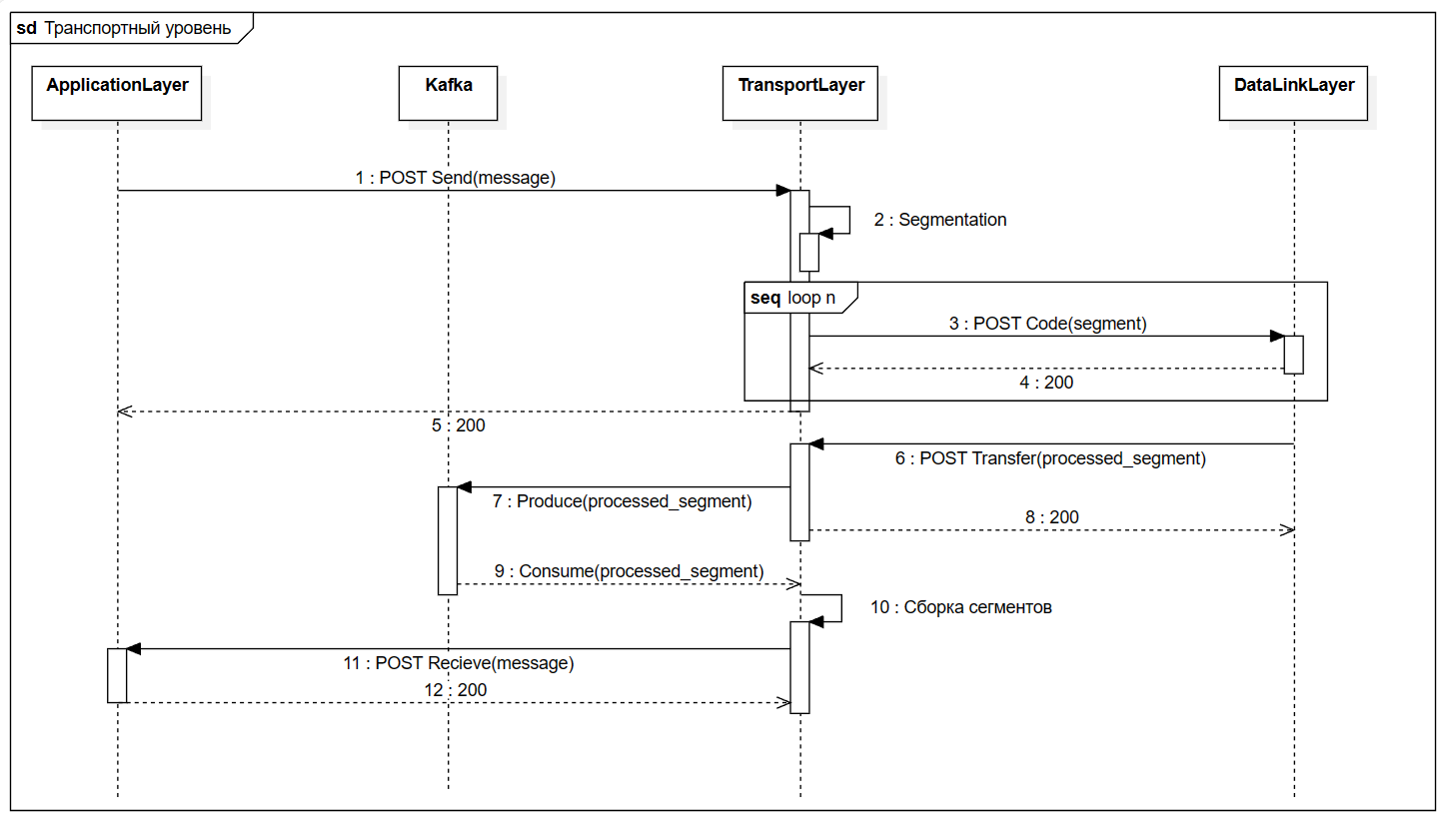


Рисунок 1 – Диаграмма последовательности транспортного уровня

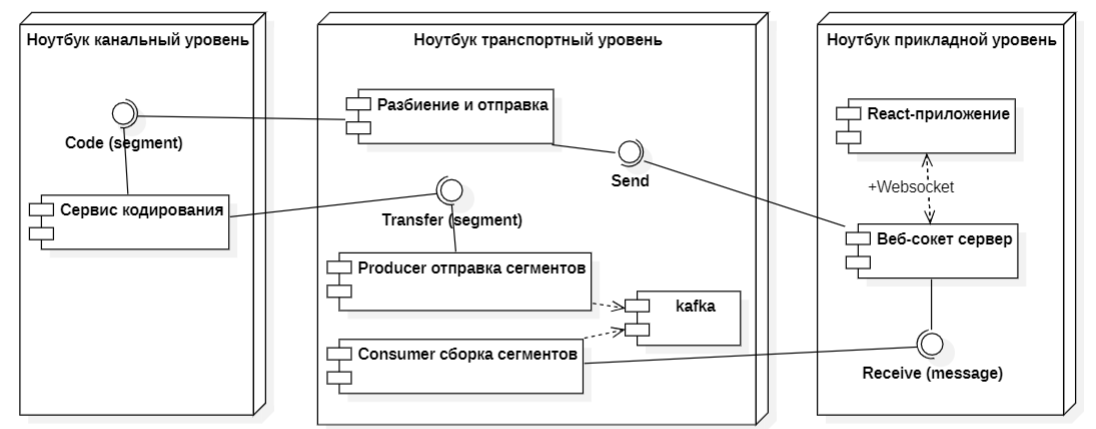


Рисунок 2 – Диаграмма развёртывания

Рассмотрим алгоритм работы транспортного уровня более подробно. Отправной точкой является вызов метода Send прикладным уровнем, в теле запроса в метод передаются имя отправителя, закодированное в строчный формат изображение и метка времени в качестве идентификации сообщения. Далее сообщение сегментируется по 2300 байт, где каждый сегмент имеет полезную нагрузку, метку времени, номер сегмента и общее число сегментов, 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.

Важной характеристикой транспортного уровня модели OSI является контроль перегрузки (congestion control). Контроль перегрузки – это различные алгоритмы, которые обеспечивают наиболее быструю скорость передачи данных между двумя узлами, передающими данные через TCP. Они управляют размером TCP-окна и могут ориентироваться на RTT (Round Trip Time — время от отправки запроса до получения ответа), потерю пакетов, время ожидания отправки пакета из очереди и т.д. [7] В текущей упрощённой реализации транспортного уровня используется механизм очереди отправки пакета. Полученный с канального уровня сегмент помещается в Kafka. Kafka – это распределенный брокер сообщений, представляющий собой отказоустойчивые конвейеры, работающие по принципу «публикация/подписка» и позволяющие обрабатывать потоки событий [8]. Из очереди раз в 2 секунды они собираются в единое сообщение и передаются на прикладной уровень с вызовом метода Receive. В случае, когда за 2 цикла часть сегментов не была принята, то сообщение передаётся на прикладной уровень с признаком ошибки.

Методы, реализуемые транспортным уровнем, представлены в таблице 1:

Таблица 1 – Методы транспортного уровня

|  |  |  |
| --- | --- | --- |
| Метод | Входные параметры | Описание |
| POST Send | time integer  sender string  payload string | Отправка сообщения с прикладного уровня на транспортный |
| POST Transfer | time integer  payload string  total integer  number integer  error boolean | 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 кодом Хэмминга [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.

Код Хэмминга — самоконтролирующийся и самокорректирующийся код. Построен применительно к двоичной системе счисления. Позволяет исправлять одиночную ошибку и находить двойную. Другими словами, это алгоритм, который позволяет закодировать какое-либо информационное сообщение определённым образом и после передачи (например, по сети) определить появилась ли какая-то ошибка в этом сообщении (к примеру, из-за помех) и, при возможности, восстановить это сообщение [6]. В коде Хэмминга источник кодирует сообщение, добавляя в сообщение избыточные биты (контрольные биты или же биты четности). Эти избыточные биты в основном вставляются и генерируются в определенных позициях сообщения для выполнения процесса обнаружения и исправления ошибок, а именно на позиции, которые соответствуют степеням двойки. Метод кода Хэмминга эффективен в сетях, где в потоках данных заданы однобитовые ошибки, еще одним преимуществом является простота использования кодов Хэмминга, что делает их наиболее подходящими для использования в компьютерной памяти и исправления одиночных ошибок. Главный недостаток — если обнаружена ошибка в нескольких битах, результатом может быть изменение другого бита, который должен быть правильным. Это может привести к дальнейшим ошибкам в данных. А для того, чтобы алгоритм работал, нужно передавать избыточные данные. Например, для 8 битного числа длина избыточных данных – 4 бита, а это, в свою очередь, 50% от самого сообщения. Для 16 же бит длина избыточных данных составляет 5 бит.

Канальный уровень получает от транспортного сегмент размером 2300 байт 2300 байт, вероятность внесения ошибки составляет 11%, а вероятность потери кадра 1%.

Для реализации данного уровня был использован golang и веб-фреймворк gin. Для решения вышеописанной задачи был создан сервер, который работает на порту 5000 и эндпоинт «code», а также был реализован пакет для работы с битовой информацией, ее кодированием, декодированием.

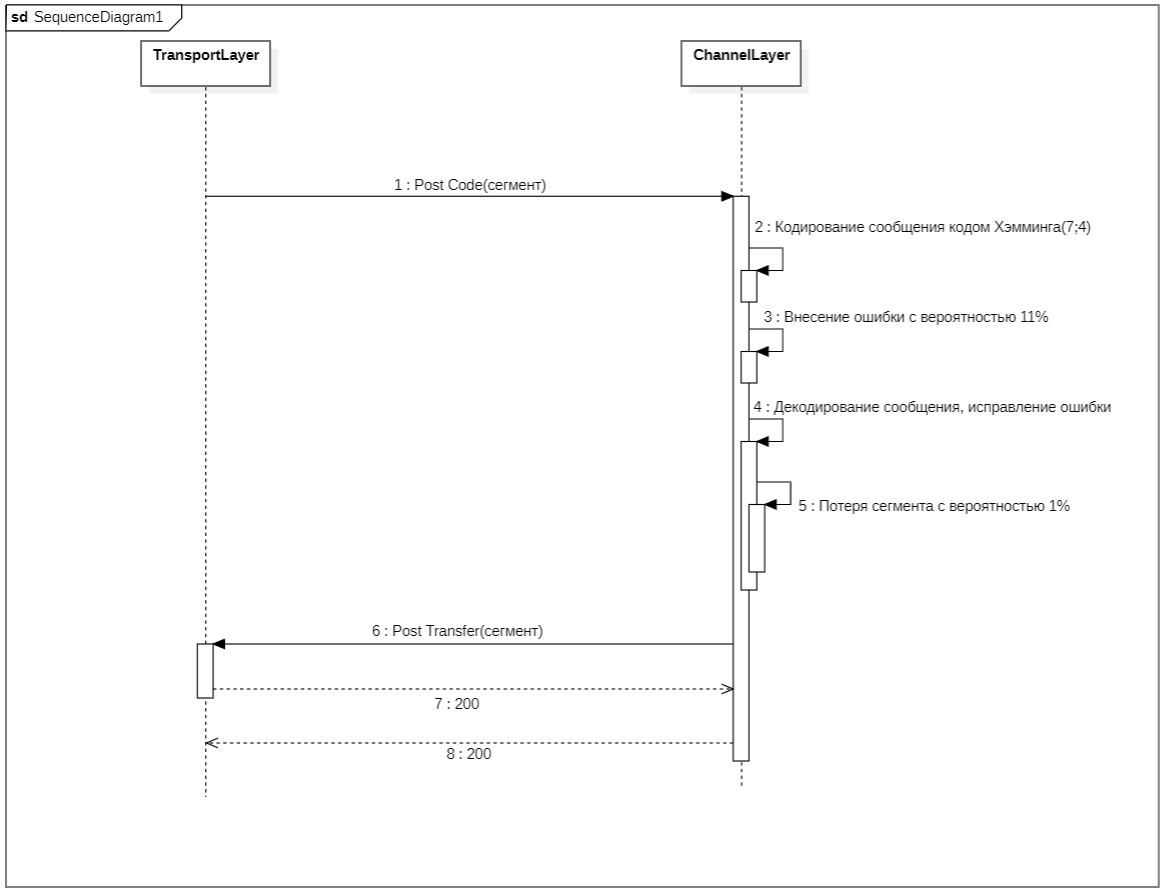
****

Рисунок 3 – Диаграмма последовательности канального уровня

Метод, используемый на канальном уровне (Таблица 2):

Таблица 2 – Методы канального уровня

| Метод | Входные параметры | Описание |
| --- | --- | --- |
| POST Code | id string  payload string  totalLength integer  segmentIndex integer | 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, то файл игнорируется, а чате отображается значок ошибки.

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

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

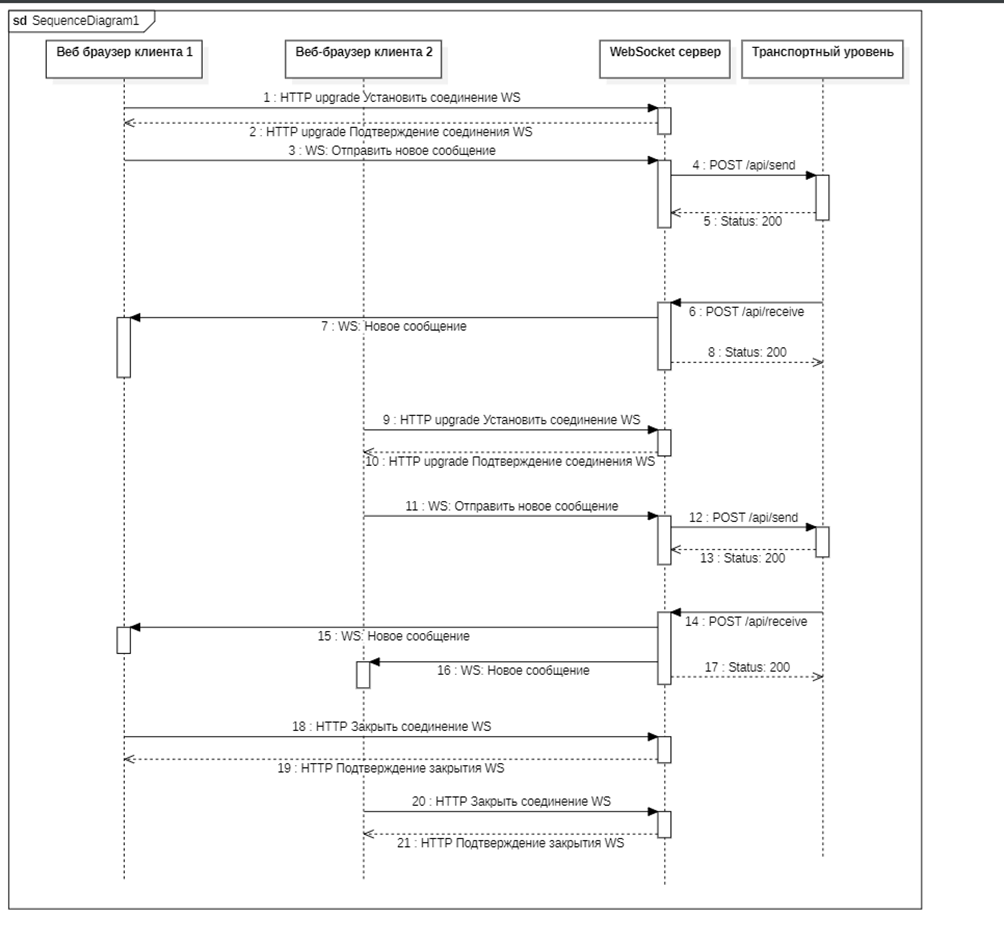


Рисунок 4 – Диаграмма последовательности прикладного уровня

Таблица 3 – Методы прикладного уровня

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

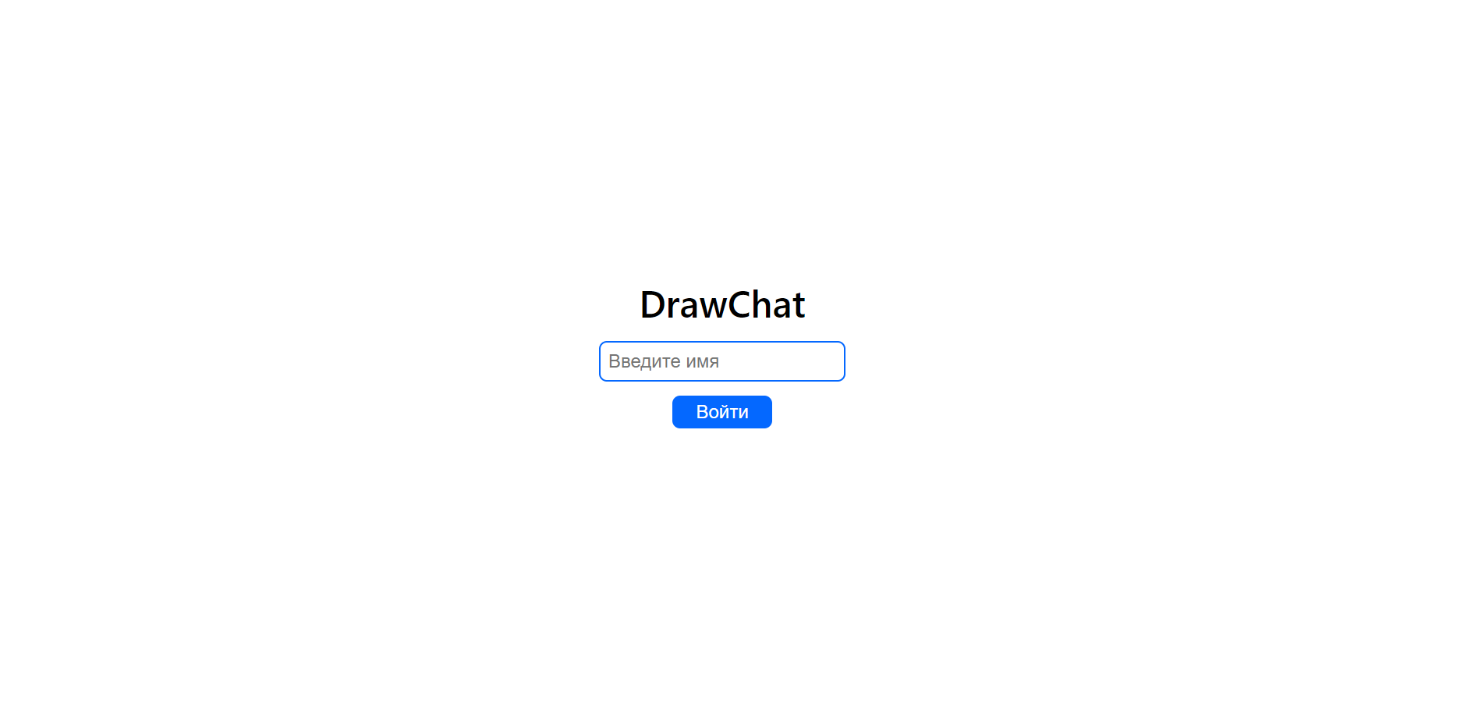


Рисунок 5 – Скриншот работы программы (авторизация)

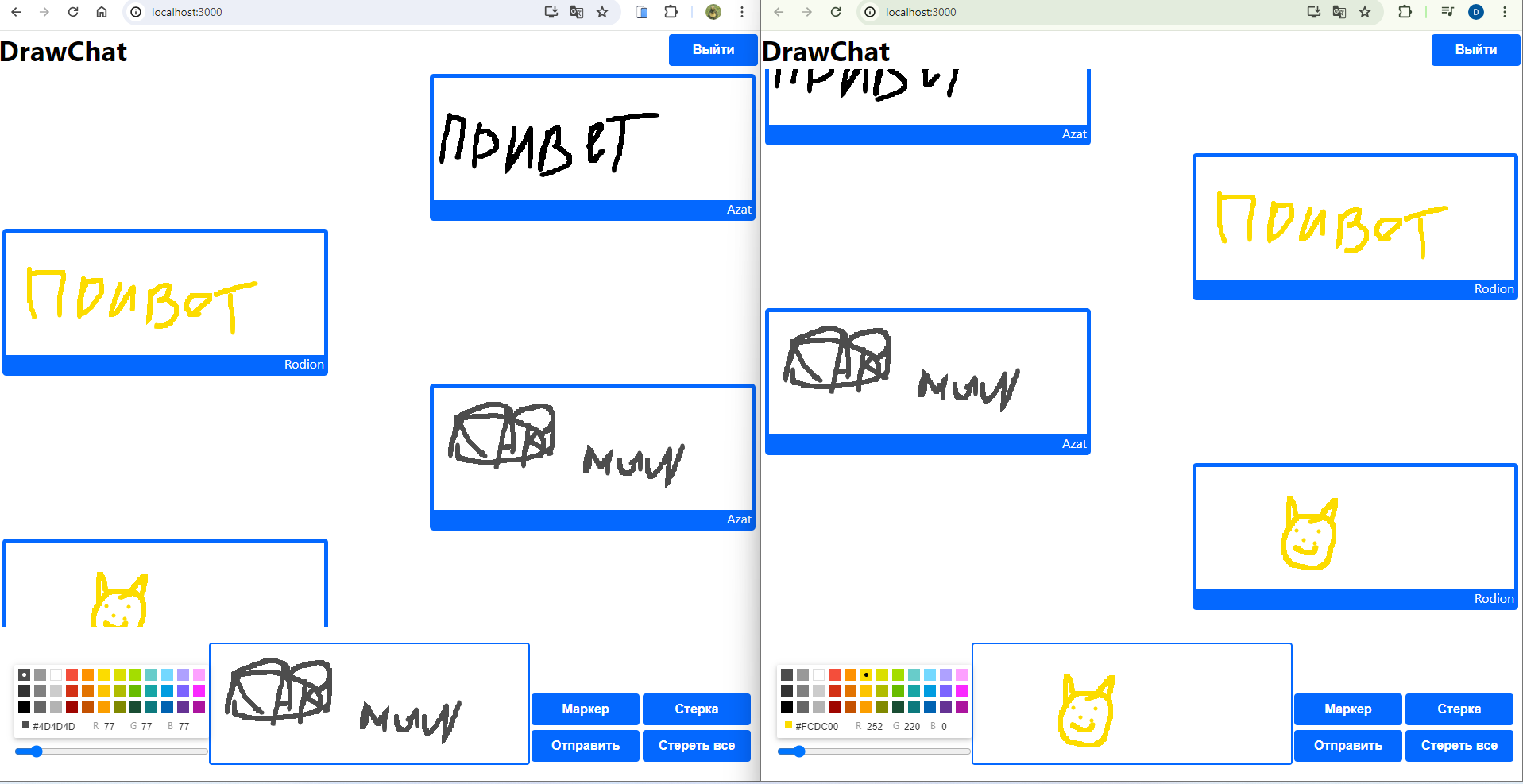


Рисунок 6 – Скриншот работы программы

# **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 20.04.2024).
  2. Полное практическое руководство по Docker [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/310460/ (дата обращения: 20.04.2024).
  3. Руководство по React [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/ruvds/articles/428077/ (дата обращения: 20.04.2024).
  4. Quick Start – React [Электронный ресурс] // React. URL: https://react.dev/learn (дата обращения: 20.04.2024).
  5. An Introduction to Programming in Go [Электронный ресурс] // Go Resources. URL: https://www.golang-book.com/books/intro (дата обращения: 20.04.2024).
  6. Код Хэмминга. Пример работы алгоритма [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/140611/ (дата обращения: 20.04.2024).
  7. TCP Congestion Control или почему скорость прыгает [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/168407/ (дата обращения: 20.04.2024).
  8. Apache Kafka: обзор [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/piter/articles/352978/ (дата обращения: 20.04.2024).

**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

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

"\_\_\_"\_\_\_\_\_\_\_\_\_2024г.

**Факультет «Информатика и системы управления»**

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

Дисциплина «Сетевые технологии в АСОИУ»

Техническое задание

Вариант 31

Студенты группы РТ5-61Б:

Билалов А.К.

Викулин И.С.

Толкачев Р.М.

2024г.

1. **Наименование:**

Распределённая система обмена цифровыми артами.

1. **Основание для разработки:**

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

1. **Исполнители:**

Билалов А. К. (транспортный уровень) – группа РТ5-61Б

Викулин И. С. (канальный уровень) – группа РТ5-61Б

Толкачев Р. М. (прикладной уровень) – группа РТ5-61Б

1. **Цель разработки:**

Разработать распределенную систему с общим чатом по обмену файлами изображений небольшого размера, в которой каждое сообщение проходит сервисы транспортного и канального уровней, разбиваясь на сегменты по 2300 байт и кодируясь [7,4]-кодом Хэмминга, и доходит до получателя с вероятностью ошибки в 11% и вероятностью потери 1%. Интерфейс пользователя соответствует дизайну сайта cian.ru.

1. **Функциональные требования:**
   1. Прикладной уровень:
      1. Разработка страницы авторизации:

* На странице авторизации пользователю предлагается ввести свой логин (имя) для входа в систему;
* Доступ к странице имеют только неавторизованные пользователи.
  + 1. Разработка главной страницы:
* Авторизованному пользователю доступен общий чат и холст для рисования изображений с кнопками «Маркер», «Стерка», «Стереть все» и «Отправить»;
  + 1. Разработка модуля Websocket для фронтенда:
* Устанавливать Websocket-соединение в момент авторизации пользователя и разрывать по нажатию кнопки «Выйти»;
* Реализовать передачу по Websocket новое сообщение с изображением в виде закодированного текста base64.
  + 1. Разработка бэкенда:
* Устанавливать Websocket-соединение для всех пользователей, прошедших авторизацию;
* Вести учет подключений и сохранять логины (имена) подключенных пользователей;
* Разработать метод send, по которому полученное сообщение передаётся на транспортный уровень с логином пользователя, временем отправки и полезной нагрузкой (изображением).
* Разработать извлечение данных (логина отправителя, времени отправки, полезной нагрузки, признака ошибки) по методу Receive и формирование сообщение, которое по Websocket широковещательно передаётся на фронтенд.
  1. Транспортный уровень:
     1. Реализация HTTP-метода Send для сегментирования сообщений:
        1. Разбиение сообщения на сегменты по 2300 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;
  2. Канальный уровень:
     1. Сервис канального уровня эмулирует канал связи с потерями:
        1. Сервис должен вносить ошибку с вероятностью 11% в один случайный бит каждого сформированного кадра;
        2. Сервис должен терять передаваемый кадр с вероятностью 1%;
     2. Реализация HTTP-метода Code для кодирования и декодирования полученного от транспортного уровня сегмента:
        1. Полученный от транспортного уровня json сегмента кодируется [7,4]-кодом Хэмминга для получения кадра;
        2. После внесения ошибки в кадр он декодируется с исправлением ошибки и передается далее в виде сегмента на транспортный уровень;

1. **Требования к составу технических средств:**

6.1. Прикладной уровень

6.1.1. Серверный часть

6.1.1.1. ПК с ОС Windows (7 и выше)

6.1.1.1. Node.js (20.11.0 и выше)

6.1.2. Клиентская часть

6.1.2.1. ПК с ОС Windows (7 и выше)

6.1.2.2. Веб-браузер Chrome (120 и выше)

6.2. Транспортный уровень

6.2.1. ПК с ОС Windows (7 и выше)

6.2.2. Bun Elysia (1.21 и выше)

6.2.3. Kafka (2.12 и выше)

6.3. Канальный уровень

6.2.1. ПК с ОС Windows (7 и выше)

6.2.2. Golang (1.21 и выше)

1. **Этапы разработки:**
   1. Разработать черновое ТЗ – 3 неделя;
   2. Разработать чистовое ТЗ, макет figma (https://www.figma.com/file/3YPQpoYupR8riAjnN73oMx/DrawChat?type=design&node-id=0%3A1&mode=design&t=wLOcs1P7packMLdj-1), диаграмму прецедентов, ER-диаграмму, диаграмму развёртывания и настроить Swagger – 8 неделя;
   3. Разработать полный комплект документов – 12 неделя;
   4. Разработать и отладить приложение – 15 неделя.
2. **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

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

"\_\_\_"\_\_\_\_\_\_\_\_\_2024г.

**Факультет «Информатика и системы управления»**

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

Дисциплина «Сетевые технологии в АСОИУ»

Программа и методика испытаний

Вариант 31

Студенты группы РТ5-61Б:

Билалов А.К.

Викулин И.С.

Толкачев Р.М.

2024г.

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. Нарисуйте сообщение и нажатие на кнопку «Отправить». | Отображение имени пользователя голубым цветом, текста сообщения и времени отправки сообщения. |
| 4 | 5.1.2.3 | Получение сообщения от другого пользователя. | Отображение имени отправителя черным цветом, изображения сообщения и времени отправки сообщения. |
| 5 | 5.1.2.4 | Рисование сообщения. Нажатие на кнопку «Маркер» | Отображение результата рисования в специальном поле |
| 6 | 5.1.2.5 | Стирание (изменения) сообщения. Нажатие на кнопку «Стёрка» | Отображение результата рисования в специальном поле |
| 7 | 5.1.2.6 | Стирание всего сообщения. Нажатие на кнопку «Стереть все» | Отображения пустого поля в специальном поле |
| 8 | 5.1.2.7 | Изменения размера маркера. Передвижение ползунка | Изменяется размер маркера в специальном поле для рисования |
| 9 | 5.1.2.8 | Изменения цвета маркера. Нажатие на цвете в палитре | Изменяется цвет маркера в специальном поле для рисования |
| 10 | 5.1.2.9 | Выход из чата. Нажатие на кнопку «Выйти». | Осуществляется переход на стартовую страницу. |
| 11 | 5.1.2.10 | Отправка сообщения с ошибкой. Нажатие на кнопку «Отправить». | Отображение значка ошибки и сообщения «При отправке сообщения возникла ошибка» |
| 12 | 5.1.3 | Запоминание имени для WebSocket подключения. Ввод имени и нажатие на кнопку «Войти». | Открытие WebSocket соединения, запоминание имени пользователя. |
| 13 | 5.1.4 | Метод Receive. | В json сообщения указывается отправитель, пэйлоад сообщения, время отправки, признак ошибки. Сообщение отправляется всем участникам чата. |
| 14 | 5.2.1 | Метод Send. Ввод сообщения, нажатие на кнопку «Отправить». | На транспортном уровне сообщение разбивается на сегменты по 2300 байт и посегментно отправляется на канальный уровень. В сегменте содержатся время отправки, общая длина сообщения, номер данного сегмента, полезная нагрузка. |
| 15 | 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. |
| 16 | 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

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

"\_\_\_"\_\_\_\_\_\_\_\_\_2024г.

**Факультет «Информатика и системы управления»**

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

Дисциплина «Сетевые технологии в АСОИУ»

Руководство пользователя

Вариант 31

Студенты группы РТ5-61Б:

Билалов А.К.

Викулин И.С.

Толкачев Р.М.

2024г.

* + - 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 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. Также 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/.

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).

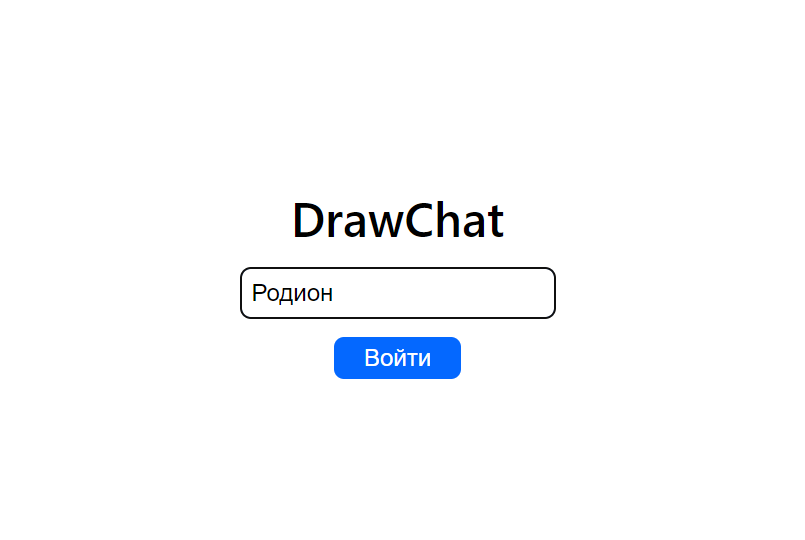


Рисунок 1 – окно идентификации

* 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).

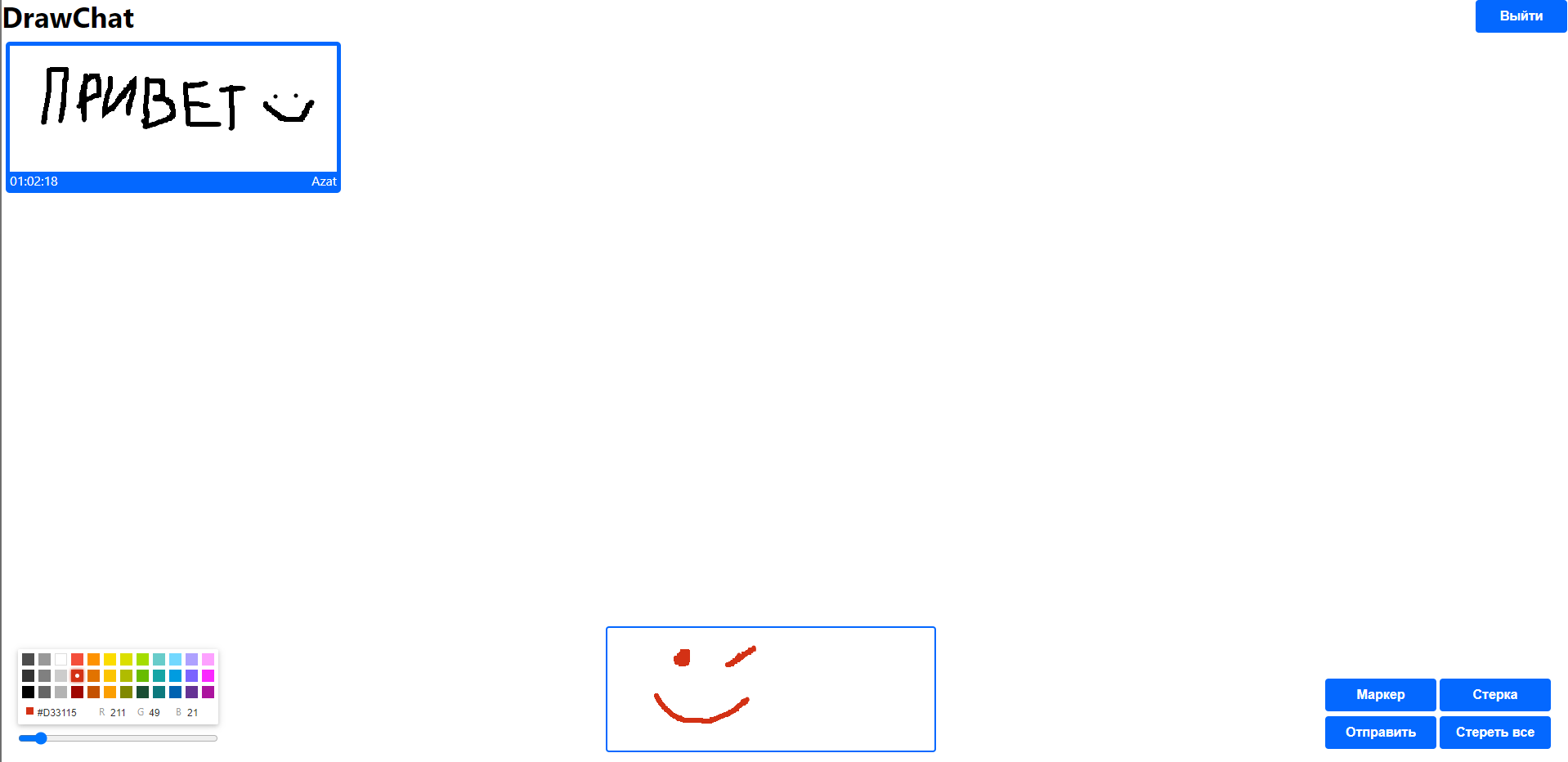


Рисунок 2 – отправка сообщения

Операция 2: для изменения цвета маркера нужно нажать на любой другой цвет палитры(рис.3).

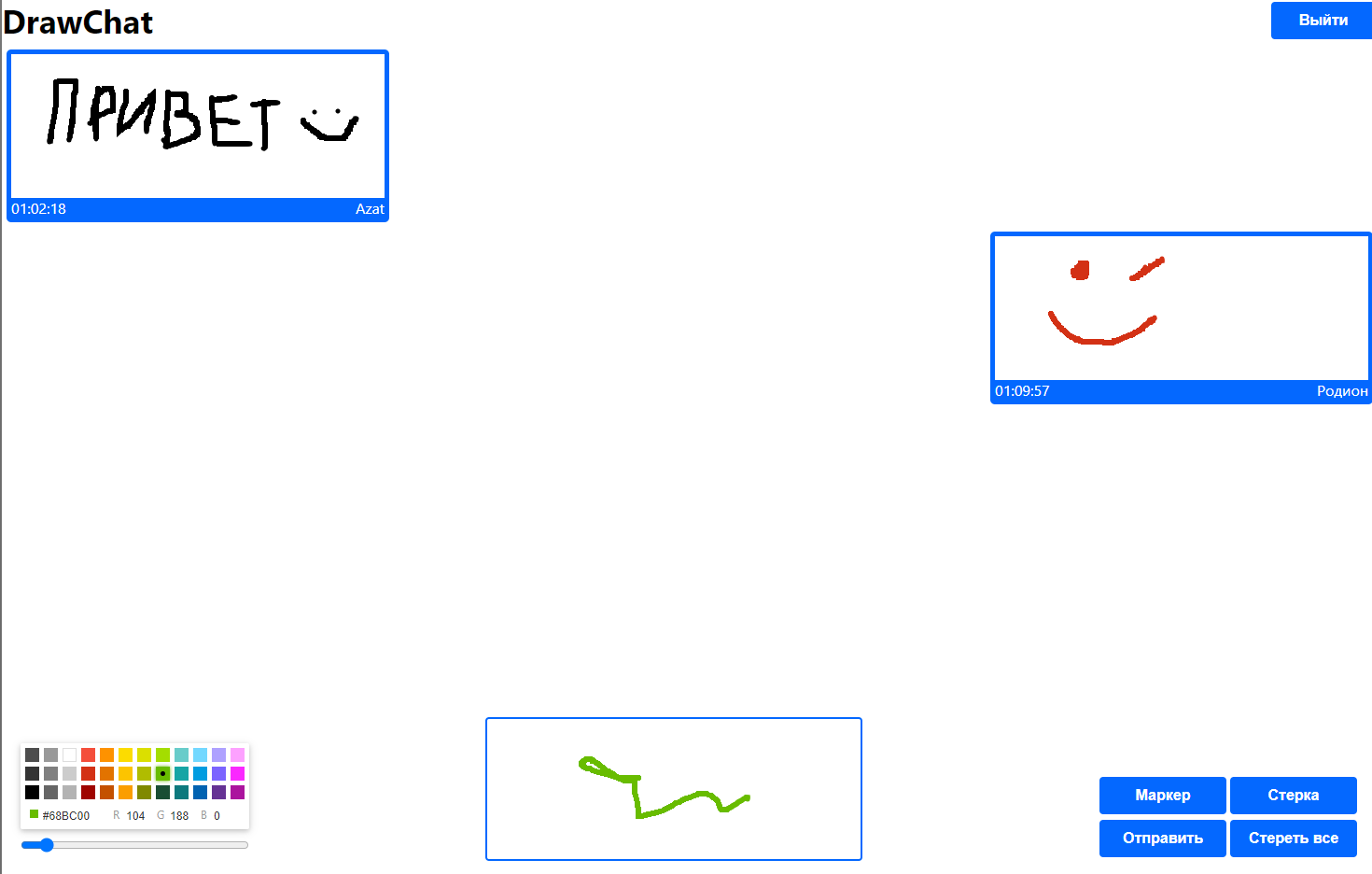


Рисунок 3 – изменение цвета маркера

Операция 3: для стирания сообщения можно нажать на кнопку «Стерка» и стирать точечно или можно нажать на кнопку «Стереть все» и полностью очистить поле. (рис.4).

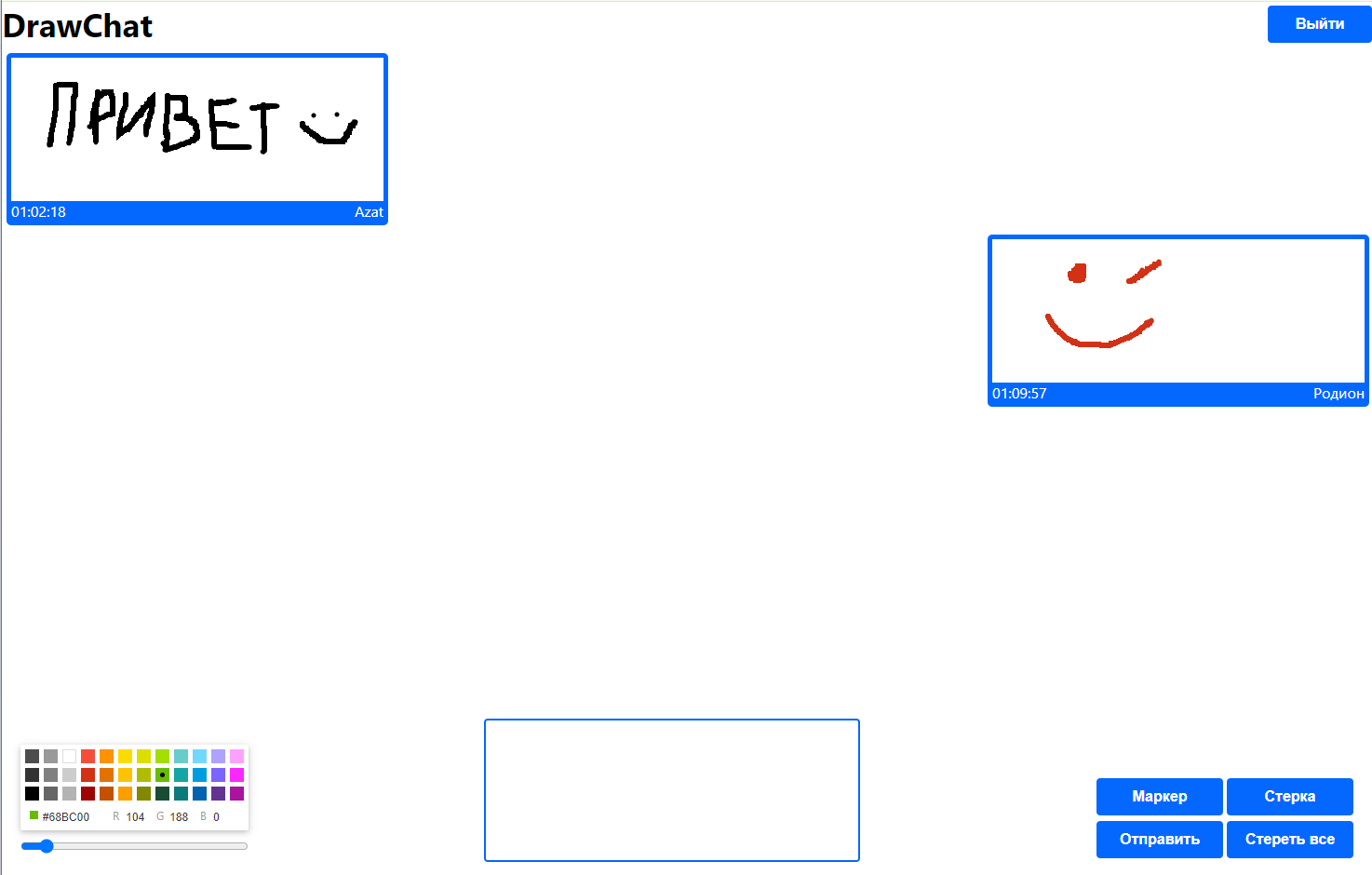


Рисунок 4 – очистка поля для ввода сообщения

Операция 4: для получения сообщений от других пользователей нужно находиться на странице чата (рис.2) и ждать сообщения. (рис.5).

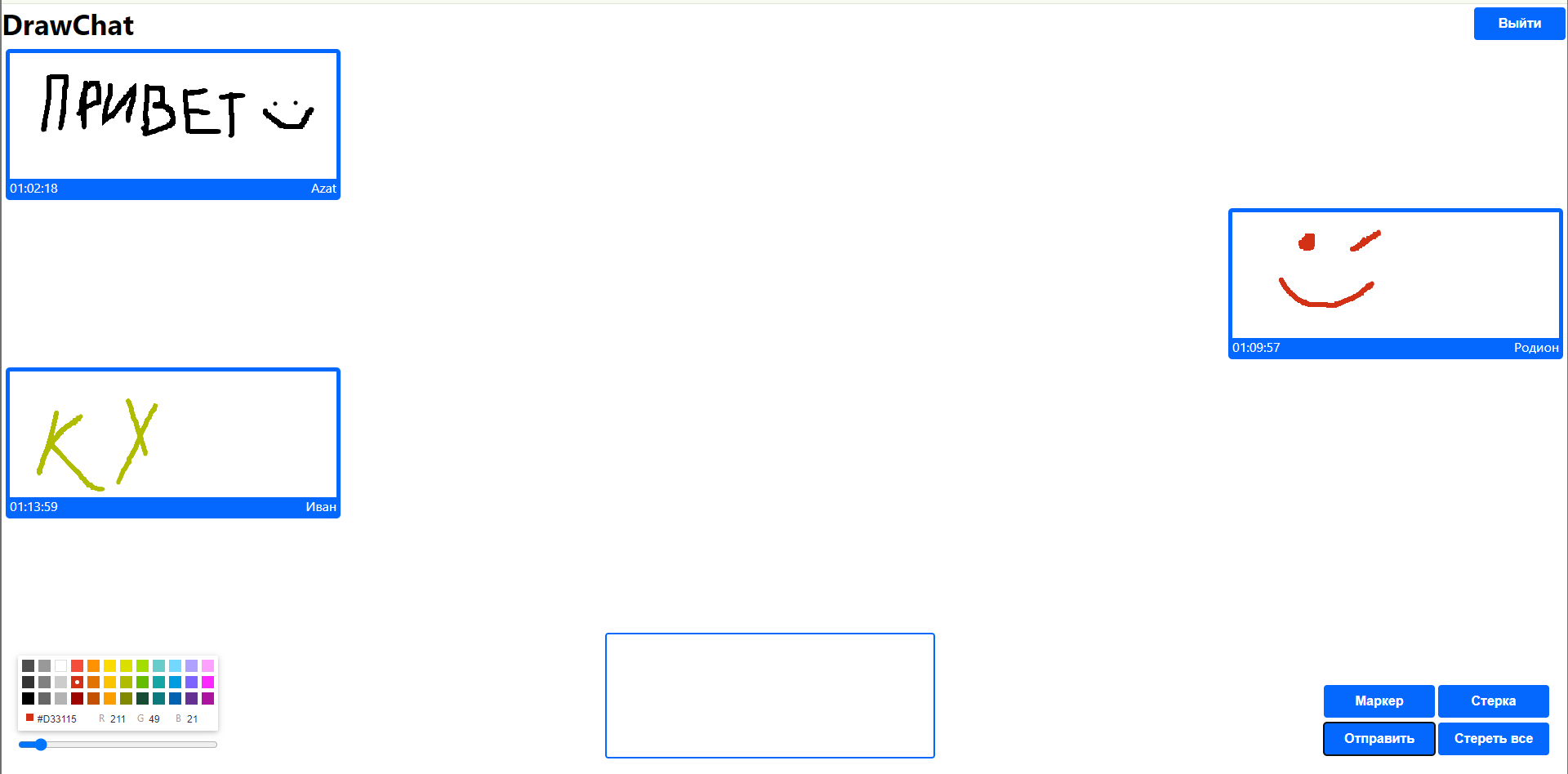


Рисунок 5 – окно чата

* 1. **Выход из чата**

Доступно для: идентифицированным пользователям.

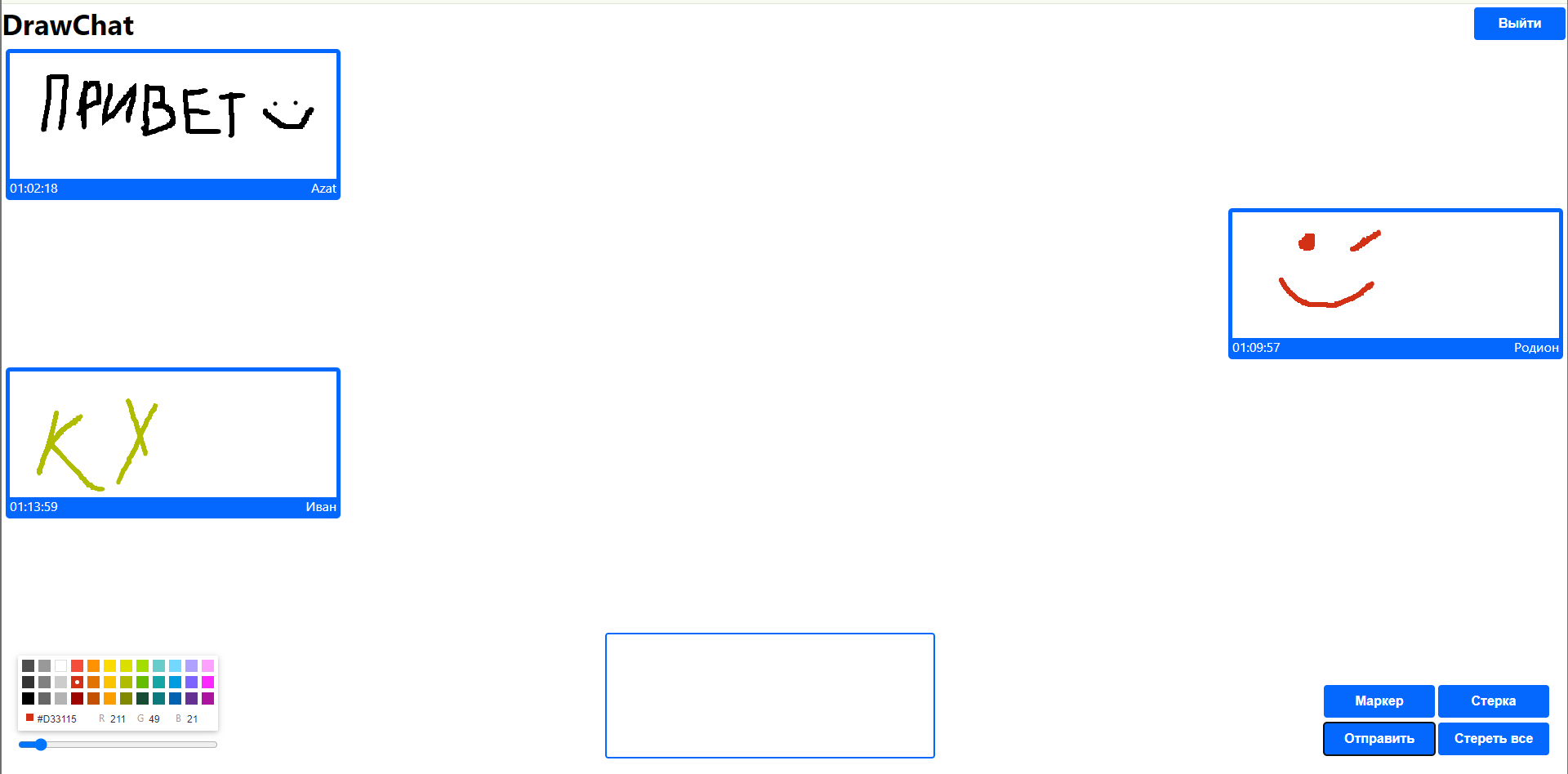
Операция: для выхода из чата необходимо нажать кнопку «выйти» (рис. 6) в правом верхнем углу страницы чата. 

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

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

"\_\_\_"\_\_\_\_\_\_\_\_\_2024г.

**Факультет «Информатика и системы управления»**

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

Дисциплина «Сетевые технологии в АСОИУ»

Руководство системного администратора

Вариант 31

Студенты группы РТ5-61Б:

Билалов А.К.

Викулин И.С.

Толкачев Р.М.

2024г.

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 (20.10 и выше)
   2. Транспортный уровень:
      1. ПК с ОС Windows (7 и выше)
      2. Node.js (20.10 и выше)
      3. Docker (23 и выше)
   3. Канальный уровень:
      1. ПК с ОС Windows (7 и выше)
      2. Golang (1.21 и выше)
2. **Инструкция по развертыванию системы:**
   1. Прикладной уровень:
      1. Клонирование репозитория:  
         git clone https://github.com/DeDeimos/draw  
         cd draw
      2. Установка зависимостей сервера:  
         cd server  
         npm install
      3. Запуск сервера:  
         node index.mjs
      4. Установка зависимостей клиента:  
         cd ../client  
         npm install
      5. Запуск пользовательского приложения:  
         npm run start
   2. Транспортный уровень:
      1. Клонирование репозитория:  
         git clone https://github.com/Azat-Bilalov/network\_transport\_level  
         cd network\_transport\_level
      2. Развёртывание контейнера:  
         docker compose up
      3. Установка зависимостей:  
         npm install
      4. Сборка проекта:  
         npm run build
      5. Запуск проекта  
         node dist/index.js
   3. Канальный уровень:
      1. Клонирование репозитория:   
         git clone <https://github.com/Vanv1k/channel-level>
      2. Запуск файла с расширением .exe