|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | Министерство науки и высшего образования Российской Федерации  **Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования**  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_ ***ИУК «Информатика и управление»\_\_\_\_\_\_\_\_\_***

**КАФЕДРА** \_\_ ***ИУК5 «Системы обработки информации»***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***Многопользовательский чат.***

по дисциплине ***Архитектура автоматизированных систем обработки информации и управления.***

Студент гр.ИУК5-52Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( *Дорофеев И. К.* )

(подпись) (Ф.И.О.)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( *Смирнов М.Е.* )

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2021

Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_(Е.В. Вершинин)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Архитектура автоматизированных систем обработки информации и управления***

Студент Дорофеев И.К. ИУК5-52Б

(фамилия, инициалы, индекс группы)

Руководитель: Смирнов М.Е.

(фамилия, инициалы)

График выполнения проекта: 25% к\_4\_нед., 50% к\_7\_нед., 75% к\_10\_нед., 100% к\_14\_нед.

***1. Тема курсового проекта***

***Многопользовательский чат***

***2. Техническое задание***

*Разработать многопользовательский чат\_\_\_\_\_\_\_\_\_\_\_\_\_*

***3. Оформление курсового проекта***

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

3.2. Перечень графического материала КП (плакаты, схемы чертежи и т. п.) \_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

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

Дата выдачи задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2021г.

(подпись) (Ф.И.О.)

Примечание:

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

СОДЕРЖАНИЕ

[1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 6](#_Toc90023177)

[1.1 Наименование 6](#_Toc90023178)

[1.2 Общие сведения 6](#_Toc90023179)

[1.2.1 Наименования предприятий разработчика и заказчика 6](#_Toc90023180)

[1.2.2 Перечень документов, на основании которых создается система 6](#_Toc90023181)

[1.2.3 Плановые сроки начала и окончания работы по созданию системы 6](#_Toc90023182)

[1.1.4 Сведения об источниках и порядке финансирования работ 6](#_Toc90023183)

[1.3 Назначение и цели создания (развития) системы 6](#_Toc90023184)

[1.3.1 Назначение системы 6](#_Toc90023185)

[1.3.2 Цели создания системы 7](#_Toc90023186)

[1.4 Характеристики объекта 8](#_Toc90023187)

[1.5 Требования к системе 8](#_Toc90023188)

[1.5.1 Требования к системе. 8](#_Toc90023189)

[1.6 Состав и содержание работ по созданию системы 9](#_Toc90023190)

[1.7 Порядок контроля и приемки системы. 10](#_Toc90023191)

[1.7.1 Состав, объем и методы испытаний системы и ее составных частей. 10](#_Toc90023192)

[1.7.2 Общие требования к приемке работ. 10](#_Toc90023193)

[1.8 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие. 10](#_Toc90023194)

[1.9 Требования к документированию 10](#_Toc90023195)

[1.10 Источники разработки 11](#_Toc90023196)

[2. НАУЧНО – ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 12](#_Toc90023197)

[Постановка задачи проектирования 12](#_Toc90023198)

[Описание предметной области 12](#_Toc90023199)

[Анализ аналогов и прототипов 13](#_Toc90023200)

[Перечень задач подлежащих решению в процессе разработки 21](#_Toc90023201)

[Обоснование выбора инструментов и платформы для разработки 22](#_Toc90023202)

[3. ПРОЕКТНО – КОНСТРУКТОРСКАЯ ЧАСТЬ 26](#_Toc90023203)

[3.1 Разработка функциональных возможностей приложения 26](#_Toc90023204)

[4. ПРОЕКТНО – ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 33](#_Toc90023205)

[5. ЗАКЛЮЧЕНИЕ 42](#_Toc90023206)

[6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТУРАТУРЫ 43](#_Toc90023207)

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

## 1.1 Наименование

Прикладное программное обеспечения для централизованного чата с использованием базы данных.

## 1.2 Общие сведения

### 1.2.1 Наименования предприятий разработчика и заказчика

Разработчик: студент группы ИУК5-52Б Дорофеев И.К.

Заказчик: КФ МГТУ им. Баумана кафедра ИУК5.

### 1.2.2 Перечень документов, на основании которых создается система

Методические указания к выполнению курсовой работы по предмету “Базы данных”

### 1.2.3 Плановые сроки начала и окончания работы по созданию системы

Начало работы: 06.09.2021

Окончание работы: 01.12.2021

### 1.1.4 Сведения об источниках и порядке финансирования работ

Источников и порядков финансирования работ не имеется.

## 1.3 Назначение и цели создания (развития) системы

### 1.3.1 Назначение системы

В совеменном мире онлайн общение играет важную роль. Люди не всегда имеют возможность позвонить друг другу или встретится лично. Чаты, форумы, службы мгновенных сообщений - все эти возможности делают общение современного человека удобным, быстрым и простым. Преимущества такого вида общения очевидны: это и скорость обработки данных, возможность простмотра предыдущих сообщений, гибкость индивидуальной настройки. С помощью чатов люди имеют возможность и находить единомышленников и делится мыслями на большую аудиторию.

База данных разрабатывается как система для надежного и удобного хранения данных. Используется в том числе в клиент-серверной архитектуре. Базы данных позволяют быстро фильтровать данные, искать определенные экземпляры и сортировать данные с учетом параметров.

### 1.3.2 Цели создания системы

* Анализ предметной области и выделение необходимых сущностей для системы;
* Создание многопользовательского чата с хранением информации в базе данных.
* разработка бизнес-логики, которая будет реализовывать эффективное и грамотное взаимодействие с входными и выходными данными;
* овладение первичными навыками ведения научно-исследовательской, проектной и производственно-технологической деятельности, развитие творческих навыков;
* подготовка к выполнению выпускной квалификационной работы;
* усвоение методов грамотного ведения, оформления и редактирования технической документации;
* исследование предметной области, изучение принципов работы с базами данных;
* тестирование разработанного программного обеспечения.

Целью курсовой работы является формирование практических навыков в области проектирования, программной реализации и анализа сложных структур данных и алгоритмов их обработки.

## 1.4 Характеристики объекта

Чат — средство обмена сообщениями по компьютерной сети в режиме реального времени, а также [программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), позволяющее организовывать такое общение. Характерной особенностью является коммуникация именно в реальном времени или близкая к этому

## 1.5 Требования к системе

### 1.5.1 Требования к системе.

Система должна быть централизованной и иметь сущности, необходимые для работы и выделенные при анализе предприятия и определения того, что нужно автоматизировать. Система должна быть устойчивой к чрезвычайным ситуациям, таким как:

* Случайное или предумышленное удаление базы данных;
* Случайное или предумышленное изменение модели данных.

Система должна предоставлять удобный и постоянный доступ к просмотру, редактированию и удалению данных.

#### 1.5.1.1 Требования к структуре и функционированию системы

Система должна иметь эргономичный интерфейс, который позволит людям любой квалификации пользоваться приложением.

#### 1.5.1.2 Требования к надежности

Программа должна обеспечивать корректную обработку исключительных ситуаций, вызванных вводом недопустимых дат, и любых других неверных данных.

#### 1.5.1.3 Требования к функциям (задачам), выполняемым системой

Система должна выполнять следующий ряд функций:

* Хранение, изменение и управление данными. Предоставление доступа к данным в любое время;
* Поиск и фильтрация данных на основе запросов к базе данных;
* Иметь защиту от повреждений в виде резервного копирования базы данных;
* Предоставлять возможность добавления чатов;
* Предоставлять возможность блокирования пользователей в отдельном чате и удаления сообщений;

## 1.6 Состав и содержание работ по созданию системы

Таблица 1. – Стадии и этапы разработки ТЗ

|  |  |  |
| --- | --- | --- |
| Стадии | Этапы работ | Сроки исполнения |
| 1. Формирование требований к АС | Исследование объекта и подтверждение  необходимости создания АС.  Формирование требований пользователя к АС.  Оформление отчёта о  выполненной работе и заявки на разработку АС. | 01.10.2021 |
| 2. Разработка концепции АС | Изучение выбранного объекта.  Проведение научно- исследовательских работ.  Проектирования концепции АС, удовлетворяющей  потребности пользователя.  Оформление отчёта о выполненной работе. | 08.10.2021 |
| 3. Техническое задание | Разработка и утверждение технического задания на  создание АС. | 22.10.2021 |
| 4. Разработка системы | Разработка системы согласно  техническому заданию. | 05.11.2021 |
| 5.Защита курсовой работы | Создание грамотной презентации и речи для защиты курсовой работы.  Защита курсовой работы. | 01.12.2021 |

## 1.7 Порядок контроля и приемки системы.

### 1.7.1 Состав, объем и методы испытаний системы и ее составных частей.

В результате предварительных испытаний на нормализацию должны быть исправлены недочёты.

Для проверки корректной работы внесённых изменений должны быть проведены повторные испытания разработанной программы.

### 1.7.2 Общие требования к приемке работ.

В процессе приемки работ должна быть осуществлена проверка на соответствие требованиям настоящего «Технического задания». По результатам испытаний возможны доработки и исправления.

## 1.8 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие.

Требований к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие не имеется.

## 1.9 Требования к документированию

По окончанию работы предъявлена расчетно-пояснительная записка, в состав которой входят:

* техническое задание;
* научно-исследовательская часть;
* проектно-конструкторская часть;
* проектно-технологическая часть.

Также должна быть предоставлена графическая часть работы, выполненная в формате А1 на 2 листах, в которую входят:

* демонстрационные чертежи;

## 1**.10 Источники разработки**

Гост 34.601-90 «Автоматизированные системы. Стадии создания»

Гост 34.602-89 «Техническое задание на создание автоматизированной системы»

# 2. НАУЧНО – ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

## Постановка задачи проектирования

Задача курсовой работы состоит в разработке программы чата, который позволяет отправлять текстовые сообщения по стредствам интернета и использования базы данных для хранения сообщений этого чата.

Необходимо создать механизм авторизации пользовательей: регистрация и авторизация, механизм выбора чата и создания новых чатов, механизм блокировки пользователя и удаления сообщений.

Необходимо разработать удобный пользовательский интерфейс для работы с программой.

## Описание предметной области

Чат ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) to chat «болтать») — средство обмена сообщениями по компьютерной сети в режиме реального времени, а также [программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), позволяющее организовывать такое общение. Характерной особенностью является коммуникация именно в реальном времени или близкая к этому, что отличает чат от [форумов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%84%D0%BE%D1%80%D1%83%D0%BC) и других «медленных» средств.

Чаты бывают централизованными и денцентрализованными. Для централизованных чатов характерна единая точка отказа. То есть отсутствие дублирования потоков отпраки данных.

Типы чатов:

1. Личные — для общения с одним собеседником. Администратор рабочего пространства не может просматривать личные чаты сотрудников.
2. Групповые — для общения с несколькими собеседниками: двумя и более. Такие чаты бывают открытыми и закрытыми.
   1. Открытые — к ним может присоединиться любой сотрудник рабочего пространства.
   2. Закрытые — в них могут участвовать только сотрудники, приглашённые администратором рабочего пространства.
3. Только для чтения (каналы) — для публикации различных новостей, только администраторы чатов могут добавлять в них сообщения.

Виды чатов:

* HTTP или веб-чаты.
* Чаты, использующие технологию Adobe Flash.
* IRC, специализированный протокол для чатов.
* Программы-чаты для общения в локальных сетях (например, Vypress Chat, Intranet Chat, Pichat).
* Чаты, реализованные поверх сторонних протоколов (например, чат, использующий ICQ).
* Чаты, работающие по схеме клиент-сервер. Например, Mychat, Jabber.
* Чаты, работающие в одно ранговых сетях. У них нет потребности в отдельном сервере, они часто используют возможности технологий DHT и TCP Relay (пример: Tox)
* Чаты, использующие технологию Push.

В совем проекте я решил использовать чат работающий по системе клиент-сервер, так как такая система наиболее подходит для приложения под Windows и проста в реализации. Также чат будет групповой(в один чат может подключаться двое и более людей).

Ориентирован для общения по средствам Интернет. Может быть интегрирован в более сложные многопользовательсие системы для общения пользователей, такие как бизнес програмы, месседжеры, социальные сети.

## Анализ аналогов и прототипов

Сегодня повсеместное использование многопользовательских чатов существенно облегчает работу человека в самых различных сферах. Впрочем, представить себе жизнь без таких приложений практически невозможно - ведь в современном мире существует много месседжеров и социальных сетей, а текже многопользовательские чаты интегрируются в почти в любой сайт или игру.

**Месседжер ICQ.**

Первым месседжером является программа ICQ. История мессенджера начинается в 1996 году, когда израильская компания Mirabilis запустила ICQ. В ней были и многопользовательские чаты, и поддержка передачи файлов, и поиск по пользователям, и много чего ещё.

Первые пользователя Интернета были в восторге от мессенджера с ярким цветком в логотипе, они помнят свой номер ICQ (UIN) еще и сегодня. В то время к UIN относились как к номеру своего телефона, а шестизначные номера ICQ стали настоящим раритетом, их продают за большие деньги на аукционе eBay. UIN был важен, прежде всего, потому, что только с его помощью можно было найти друзей. Привычной сейчас возможности пролистать «списки контактов» тогда еще не было. Именно этот принцип работы мессенджера делал атмосферу в «аське» такой непринужденной. Спустя всего лишь два года после появления ICQ (от английского: «I seek you» — «Я тебя ищу») мессенджер был куплен американским медиа-концерном AOL за 407 миллионов долларов.

Технические возможности ICQ:

* Когда вы открывали приложение, первое, что вы видели, - список, содержащий все ваши контакты и их статусы, указывающие «В сети» ли они.
* Большие мигающие иконки сообщали вам, кто в данный момент доступен, занят или отошёл.
* Звуковые уведомления позволяли узнать об изменении статуса ваших друзей.
* По умолчанию, при входе в систему вы тоже транслировали всему миру, что вы в сети.
* Что-то обсуждать с собеседником можно было, только когда он находился в сети. Были и групповые чаты, но их использование также было связано со множеством проблем и ограничений. Вы могли присоединиться к групповому чату лишь войдя в сеть, но при этом участники чата не обязательно могли быть в сети одновременно с вами. Любой человек мог быть внезапно отключен по какой-либо причине (неустойчивость соединения или телефонный звонок). Вы пропускали все разговоры, происходившие в то время, пока вы были оффлайн. Кроме того, групповые чаты были постоянно завалены автоматическими сообщениями об изменении статуса контактов: Иван присоединился, Антон отключился.

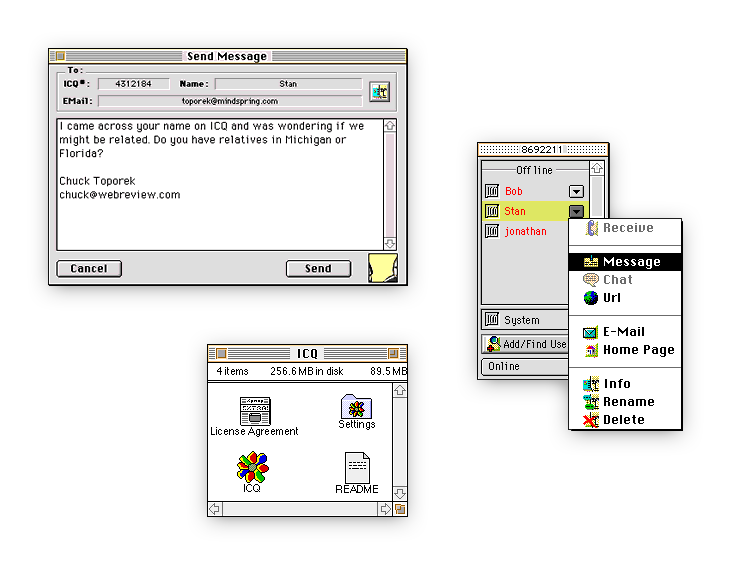


Рисунок.1 Интерфейс программы «ICQ» .

**Telegram.**

Telegram — [кроссплатформенный](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) [мессенджер](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BC%D0%B3%D0%BD%D0%BE%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BC%D0%B5%D0%BD%D0%B0_%D1%81%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8) с функциями [VoIP](https://ru.wikipedia.org/wiki/VoIP), позволяющий обмениваться текстовыми, голосовыми и видеосообщениями, стикерами и фотографиями, файлами многих форматов. Также можно совершать видео- и аудиозвонки, организовывать конференции, многопользовательские группы и каналы. Клиентские приложения Telegram доступны для [Android](https://ru.wikipedia.org/wiki/Android), [iOS](https://ru.wikipedia.org/wiki/IOS), [Windows Phone](https://ru.wikipedia.org/wiki/Windows_Phone), [Windows](https://ru.wikipedia.org/wiki/Windows), [macOS](https://ru.wikipedia.org/wiki/MacOS) и [GNU/Linux](https://ru.wikipedia.org/wiki/GNU/Linux). Количество ежемесячных активных пользователей сервиса по состоянию на январь 2021 года составляет около 500 млн человек. В августе [2017 года](https://ru.wikipedia.org/wiki/2017_%D0%B3%D0%BE%D0%B4) в своем Telegram-канале Павел Дуров заявил, что количество пользователей мессенджера ежедневно увеличивается более чем на 600 тысяч.

Telegram — [облачный](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%87%D0%BD%D0%BE%D0%B5_%D1%85%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) мессенджер, его можно использовать одновременно на нескольких устройствах, и все чаты и файлы (за исключением секретных чатов) будут доступны на этих устройствах. Во всех чатах можно использовать голосовые сообщения, видеосообщения, прикрепление файлов, стикеры, gif-анимации и [эмодзи](https://ru.wikipedia.org/wiki/%D0%AD%D0%BC%D0%BE%D0%B4%D0%B7%D0%B8); есть отметка о том, что собеседник прочитал сообщение и т. д.

**Технология:**

Для мессенджера был создан протокол [MTProto](https://ru.wikipedia.org/wiki/MTProto), предполагающий использование нескольких протоколов шифрования. При авторизации и аутентификации используются алгоритмы [RSA-2048](https://ru.wikipedia.org/wiki/RSA), [DH](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%94%D0%B8%D1%84%D1%84%D0%B8_%E2%80%94_%D0%A5%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD%D0%B0)-2048 для шифрования, при передаче сообщений протокола в сеть они шифруются [AES](https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard) с ключом, известным клиенту и серверу. С переходом на протокол MTProto 2.0 применяется криптографический [хеш-алгоритм](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [SHA-256](https://ru.wikipedia.org/wiki/SHA-2).

C 8 октября 2013 года в мессенджере появился режим «секретных» чатов (Secret Chats). Этот режим реализует шифрование, при котором лишь отправитель и получатель обладают общим ключом (end-to-end шифрование), с применением алгоритма AES-256 в режиме IGE ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Infinite Garble Extension) для пересылаемых сообщений. В отличие от обычного режима, сообщения в секретных чатах не расшифровываются сервером, история переписки сохраняется лишь на тех двух устройствах, на которых был создан чат.

Клиенты и серверы:

Клиенты для Telegram включают в себя:

* Мобильные приложения для [Android](https://ru.wikipedia.org/wiki/Android) и [iOS](https://ru.wikipedia.org/wiki/IOS)/[iPadOS](https://ru.wikipedia.org/wiki/IPadOS);
* Настольные приложения для [Windows](https://ru.wikipedia.org/wiki/Windows), [Linux](https://ru.wikipedia.org/wiki/Linux) и [macOS](https://ru.wikipedia.org/wiki/MacOS);
* Веб-приложение, веб-приложения для [Chrome app](https://ru.wikipedia.org/w/index.php?title=Google_Chrome_App&action=edit&redlink=1)[[en]](https://en.wikipedia.org/wiki/Google_Chrome_App), веб-приложение для [React](https://ru.wikipedia.org/wiki/React).

Также есть несколько неофициальных клиентов, в том числе с открытым исходным кодом, в том числе под лицензией [GNU GPL](https://ru.wikipedia.org/wiki/GNU_General_Public_License).

Используются [проприетарная](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D0%B5%D1%82%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) серверная часть c [закрытым кодом](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), работающая на мощностях нескольких компаний в США, Нидерландах, и в Сингапуре.

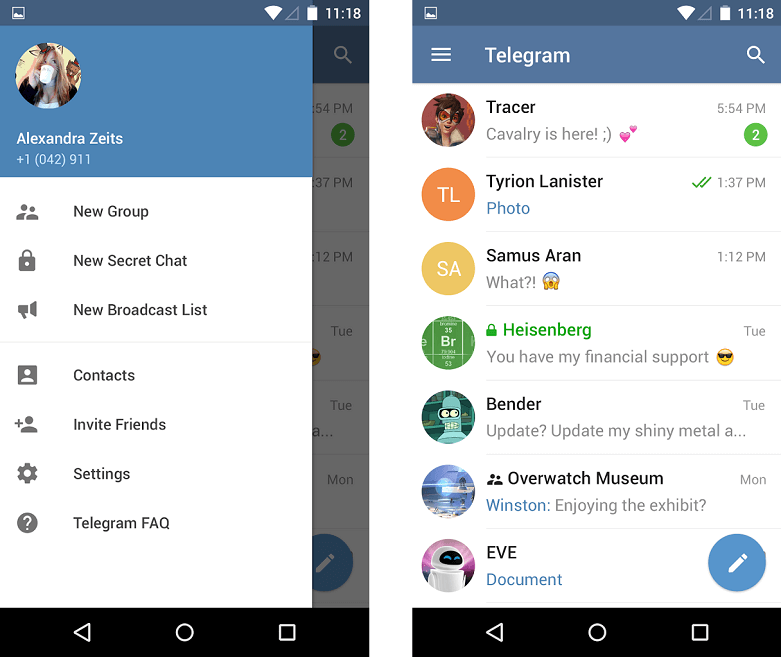


Рисунок.2 Интерфейс программы «Telegram».

**Rocket Chat.**

Rocket.Chat — это бесплатная, масштабируемая, гибко настраиваемая и безопасная платформа с открытым исходным кодом, которая позволяет вам общаться и сотрудничать с вашей командой, обмениваться файлами и общаться в режиме реального времени. Она является кроссплатформенной и работает в мобильных операционных системах Linux, Windows, macOS, Android и iOS.

Платформа похожа на Slack и включает в себя чат в реальном времени, бесплатные аудио- и видеоконференции, каналы, гостевой доступ, совместное использование экрана и общий доступ к файлам. Для обеспечения [безопасной связи](https://blog.sedicomm.com/2020/07/15/nagruzochnoe-testirovanie-veb-serverov-s-pomoshhyu-siege-benchmarking-tool/) поддерживает:

* групповую синхронизацию LDAP
* двухфакторную аутентификацию (2FA)
* сквозное шифрование
* единый вход
* несколько поставщиков Oauth аутентификации

Достоинства и недостатки ROCKET.CHAT:

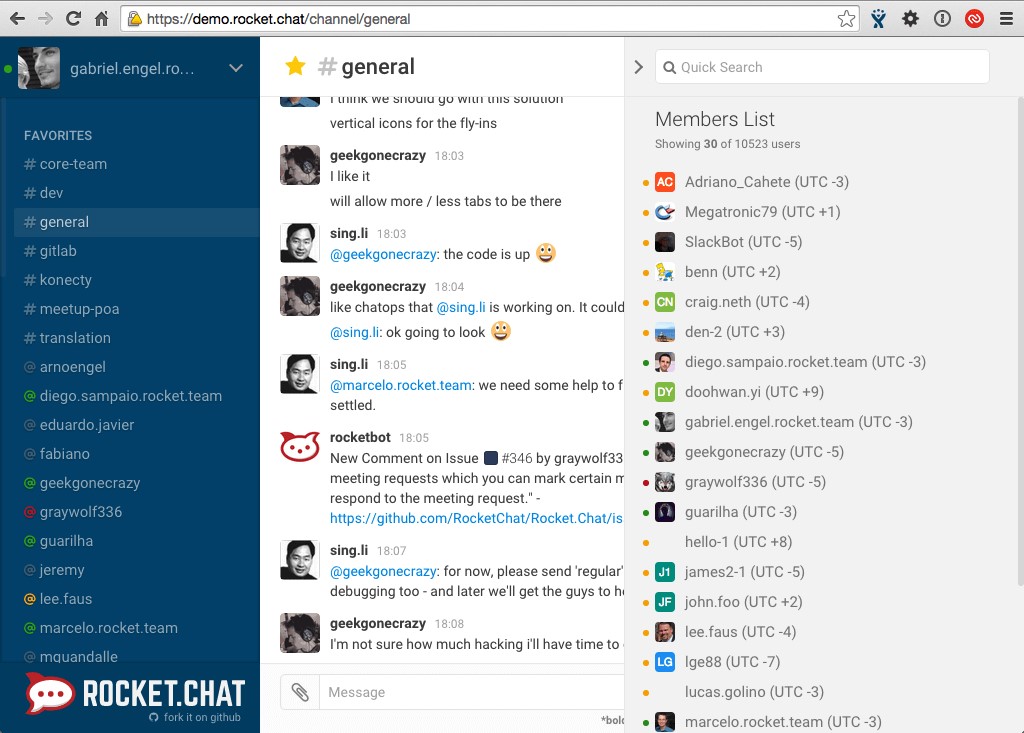
**+ Возможность установки** **Rocket.Chat** **на собственные серверы**, что делает его бесплатным и дает вам инструменты для тонкой настройки. А еще наши западные партнеры не смогут отключить вас за неуплату или, например, с вводом новых санкций.  
**+ Конфиденциальность**. Ваша переписка не утечет непонятно куда, даже если вы арендуете серверы. Rocket.Chat не предоставляет ваши данные и сообщения третьим лицам и не хранит удаленные сообщения.  
**+ Защищенность.** Статистика по взлому аккаунтов говорит не в пользу персональных мессенджеров, их компрометируют гораздо чаще. Дополнительную безопасность дает вышеупомянутая возможность установки Rocket.Chat на собственные серверы.  
**+ Приватность.** Благодаря тому, что круг контактов ограничен вашими коллегами, личные дела, реклама и спам не будут отвлекать вас. А вы, в свою очередь, не сможете «случайно поделиться» вашей перепиской с людьми, которым она не предназначается.  
**+ Открытость** **Rocket.Chat**. Будучи open source проектом, Rocket.Chat позволяет нам самостоятельно исправлять ошибки и публиковать обновления. При желании можно модифицировать приложение более, чем полностью.  
**+ Поддержка всех базовых функций.** Rocket.Chat обеспечивает стабильную работу всех основных функций, используемых мессенджерами. Этим закрываются потребности подавляющего большинства пользователей.  
**+ Возможность модификации** для расширения функционала. Например, нам позарез нужны была поддержка видеоконференций, и мы ее получили!  
**+ Постоянное развитие.** Rocket.Chat регулярно обновляется, становясь все удобнее и приятнее в использовании.  
  
МИНУСЫ:  
**- Баги.** Они все еще есть, хоть и не распространяются на базовый функционал. Из-за ограниченной популярности Rocket.Chat база тестировщиков не такая уж и обширная. В основном ошибки связаны с новыми инструментами и быстро исправляются.  
**- Необходимость первичной настройки.** Другие мессенджеры, особенно если речь идет о персональных, гораздо дружелюбнее к новичкам. Особенно если хотите развернуть свой собственный сервер. Но такая настройка – одноразовое мероприятие, которое быстро окупит себя.  
**- Слишком большой стек решений.** Для того, чтобы установить нужные надстройки, придется покопаться в обширной библиотеке, чтобы выбрать подходящие. А если вы используете специфические, узконаправленные модификации, они могут перестать работать или заглючить после обновления.  


Рисунок 3. Интерфейс программы «Rocket.Chat ».

В итоге мы можем выделить один общией минус существующих чатов – это закрытый исходный код серверной части. Мы не можем быть уверены в безопасности личных данных пользуясь данными приложениями. Поэтому я решил создать свой чат, в котором я могу гарантировать безопасность личных данных.

## Перечень задач подлежащих решению в процессе разработки

Задачи, подлежащие решению в процессе разработки, следующие:

1) соблюдение правильности отправки сообщений без потерь данных;

2) программа должна иметь простой, но в то же время понятный и наглядный интерфейс, который не должен перегружать ресурсы компьютера;

3) программа должна иметь возможность выбора никнейма пользователя;

5) программа не должна занимать большой объем памяти и не должна требовать установки на жесткий диск компьютера;

7) работоспособность приложения в среде Windows;

## Обоснование выбора инструментов и платформы для разработки

Для выполнения курсовой работы были выбраны:

1. Операционная система Windows 10.

Данная операционная система является одной из самых распространенных на данный момент, она обладает так же рядом существенных преимуществ. Одним из критериев выбора Windows 10 является то, что она установлена на многих предприятиях

Это единый (для всех программ-приложений Windows), стандартизированный и интуитивно-понятный интерфейс взаимодействия с пользователем. Т.е., даже начинающему пользователю персонального компьютера, не требуется особых усилий для освоения навыков работы с данным программным средством.

Причины выбора Операционной системы Windows 10:

1. Простота эксплуатации. Даже неподготовленному человеку легко разобраться в принципах работы операционной системы, ведь все интуитивно понятно, также система не требует знаний в программировании.

2. Большое количество программ сторонних компаний выпускают именно под данную операционную систему. Есть как платные, так и бесплатные версии.

3. Большая распространённость. Данная операционная система, по состоянию на апрель 2020 года, заняла второе место в общей мировой статистике использования ОС.

2. Язык программирования С#

В настоящее время C # и чаще всего используются для создания коммерческих и бизнес-приложений.

Основные преимущества языка:

1. По умолчанию C # запрещает прямое управление памятью, возвращая различные типы систем и сборку мусора. Работа напрямую с памятью все еще существует в специальном режиме кода «опасности», но это должно быть четко объявлено. По этой причине в C # активно используется только один поставщик доступа «.».
2. В C # нет методов множественного наследования. Но вместо этого предлагается использовать создания нескольких интерфейсов. Хотя многие мнения о наследовании разные. Но отсутствие этого механизма в C # должно, по крайней мере, облегчить разработку компилятора.
3. Система типов C # представляет систему типов. NET как большинство современных современных языков. Важной особенностью системы этого типа является четкое разделение всех категорий на типы значений и ссылочные типы.

C # поддерживает .NET Framework 4, что позволяет использовать опыт разработки приложений для устройств.

Исходя из всего вышеперечисленного, можно прийти к выводу, что язык программирования C#, несомненно, подходит для разработки автоматизированной информационной системы.

3. Объектно-реляционная система управления базами данных PostgreSQL

Из характеристик СУБД, которые могут определить выбор, одной из важнейших является модель данных. Теоретически любую информацию можно представить в виде реляционной модели. Сила реляционных баз данных в том, что эта модель очень хорошо подходит для предприятий, которые располагают немалыми средствами для активного внедрения передовых систем. Эта модель имеет наиболее проработанное математическое основание и хорошо проработанные стандарты. Реляционная модель данных отличается большой гибкостью с точки зрения изменения структуры данных. Здесь можно менять физическую структуру данных, не переписывая приложения.

Важным критерием выбора с точки зрения перспектив становится наличие эффективных средств разработки. Такие средства, обладающие удобным интерфейсом, позволяют специалистам предприятия самостоятельно и быстро настраивать информационные системы в соответствии с требованиями бизнеса.

PostgreSQL - это мощная объектно-реляционная система управления базами данных. Она включает большинство типов данных SQL92 и SQL99, включая integer, numeric, boolean, char, varchar, date, interval, и timestamp. Она также поддерживает хранение больших двоичных объектов (BLOB's), включая картинки, звук, или видео. А также имеет API для C# и др.

Являясь СУБД класса предприятия, PostgreSQL предоставляет такие особенности как Multi-Version Concurrency Control (MVCC), восстановление по точке во времени, табличное пространство, асинхронная репликация, вложенные транзакции (точки сохранения), горячее резервирование, планировщик/оптимизатор запросов, и упреждающее журналирование на случай поломки.

Средства обеспечения целостности данных включают составные первичные ключи, внешние ключи с поддержкой запрета и каскадирования изменений/удалений, проверку ограничений (constraints), ограничения уникальности и ограничения на непустые значения.

3. Среда разработки Microsoft Visual Studio 2019

Microsoft Visual Studio 2019— это набор инструментов разработки, основанных на использовании компонентов, и других технологий для создания мощных, производительных приложений. Кроме того, среда Visual Studio оптимизирована для совместного проектирования, разработки и развертывания корпоративных решений. Visual Studio предоставляет средства для проектирования, разработки b отладки.

Среда разработки Visual Studio представляет собой полный набор средств разработки для создания настольных приложений. Visual C# использует единую интегрированную среду разработки (IDE), которая позволяет совместно использовать средства и упрощает создание решений на базе нескольких языков.

Почему именно эта версия продукта, потому что в нем есть такие удобные нововведения как:

1. Visual Studio IntelliCode повышает эффективность разработки программного обеспечения с помощью искусственного интеллекта (ИИ). Для создания рекомендаций IntelliCode анализирует 2000 проектов с открытым кодом на GitHub
2. Рефакторинг В C#. Есть много новых удобных возможностей рефакторинга, которые помогают упорядочить код. Они отображаются как предложения со значком лампочки и включают такие действия, как перемещение элементов в интерфейс или базовый класс, настройку пространств имен в соответствии со структурой папок.

# 3. ПРОЕКТНО – КОНСТРУКТОРСКАЯ ЧАСТЬ

## 3.1 Разработка функциональных возможностей приложения

Для разработки приложения использовалась среда Microsoft Visual Studio, включающая в себя широкий набор функциональных инструментов.

Для представления принципа работы приложения ниже приведена ее

UML диаграмма вариантов использования (рисунок 3.1).

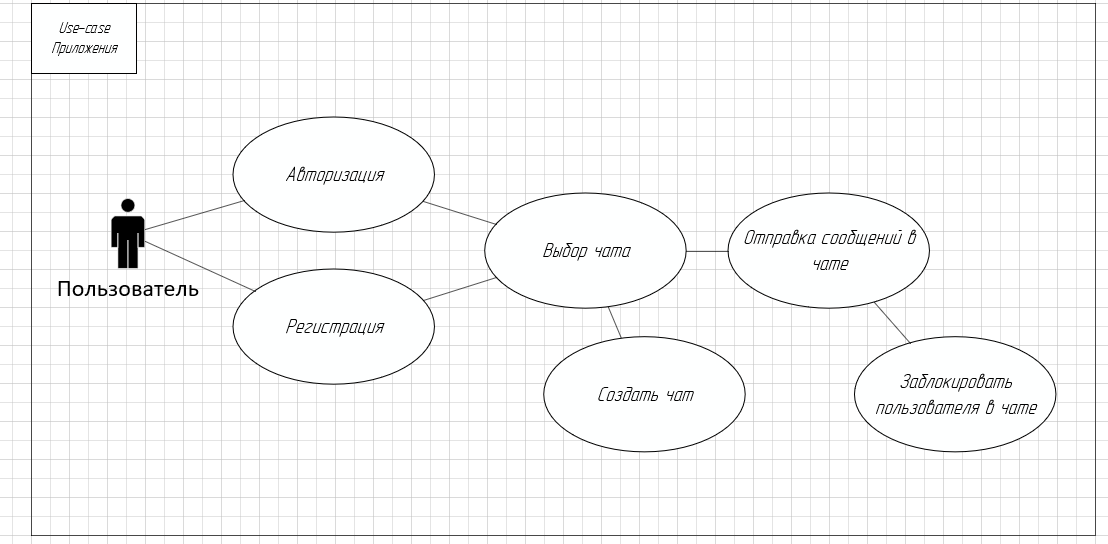


Рисунок 3.1 - Диаграмма использования веб-приложения

При запуске приложения пользователь может войти с помощью логина и пароля или зарегестрироватсья. Далее ему будет предложен список чатов или кнопка для создания собственного чата в котором он будет являться администратором с возможностью блокирования пользователей. После выбора чата будет открываться окно чата в котором из базы данных загружены предыдущие сообщения этого чата и пользователь получает возможность отправлять сообщения и просматривать список участников этого чата которые «онлайн» в данный момент. Если пользователь является администратором он может открыть окно в котором доступен механизм блокирования пользователя в данном чате.

1. **Разработка архитектуры приложения**

Архитектура приложения - клиент серверная.



Рисунок 3.1 – Схема архитектуры приложения

Когда пользователь отправляет сообщение, оно по средствам сокетов отправляется на сервер где обрабатывается и пересылается другим пользователям, которые онлайн в данный момент, а также все данные записываются в базу данных. Когда пользователь входит в чат все сообщения подгружаются из БД.

Клиент-серверная архитектура представляет собой иерархическую сеть, которая состоит из узлов-клиентов (их может быть от одного и до неограниченного количества) и центрального сервера, через который выполняется хранение и обработка данных, а так же передача их в обоих направлениях.

Клиент - это объект, запрашивающий информацию по сети. Как правило, это персональный компьютер или рабочая станция, запрашивающая информацию у сервера.

Сервер - компьютер, хранящий и обрабатывающий информацию, с которой работают сетевые клиенты.

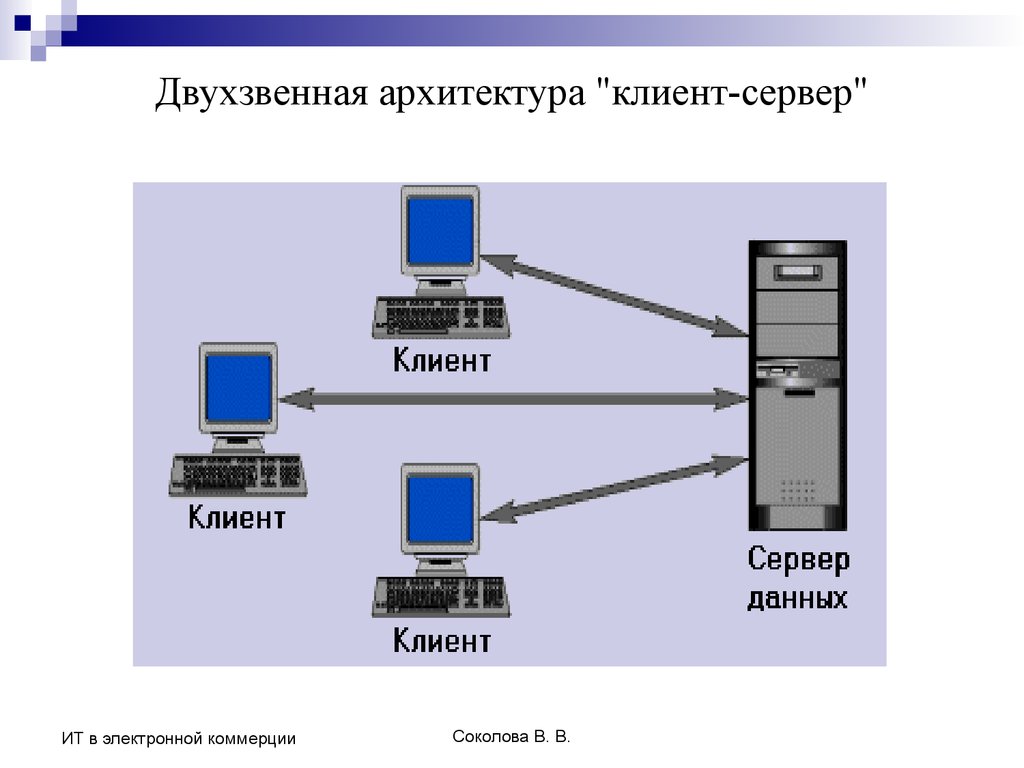


Рисунок 3.3— Принцип работы клиент серверной архитектуры приложения.

Как и любая другая технология, клиент-серверная архитектура имеет свои преимущества и свои недостатки. Рассмотрим их.

Преимущества**:**

1. Выполнение большей части работы мощной серверной частью при минимуме нагрузки на клиента.
2. Основная часть данных хранятся на сервере. При этом, как правило, он лучше защищён от различного вида угроз, чем обычный клиентский ПК.
3. Возможность более чёткого разграничения полномочий доступа к разным уровням информационной системы. Каждому клиенту — свой уровень доступа.
4. Кроссплатформенность. Проще говоря, любой клиент может работать с ресурсами сервера вне зависимости от используемой операционной системы.
5. Уменьшение нагрузки на сеть ввиду того, что клиент в основном передаёт серверу команды, а тот уже их исполняет.

Недостатки**:**

1. Выход из строя сервера может привести к неработоспособности всей системы, его использующей.
2. Высокая стоимость серверного оборудования и его обслуживания (в частности, может потребоваться отдельный специалист для обслуживания).
3. Высокая нагрузка на серверное оборудование и канал связи до него.

В случае многопользовательского централизованного чата информация с клиента отправляется на сервер, где обрабатывается и в свою очередь отправляется другим клиентам.

Основные протоколы передачи данных:

1. TCP/IP — Transmission Control Protocol/Internet Protocol

Это стек протоколов TCP и IP. Первый обеспечивает и контролирует надёжную передачу данных и следит за её целостностью. Второй же отвечает за маршрутизацию для отправки данных. Протокол TCP часто используется более комплексными протоколами.

1. UDP — User Datagram Protocol

Протокол, обеспечивающий передачу данных без предварительного создания соединения между ними. Этот протокол является ненадёжным. В нём пакеты могут не только не дойти, но и прийти не по порядку или вовсе продублироваться.

Основное преимущество UDP протокола заключается в скорости доставки данных. Именно поэтому чувствительные к сетевым задержкам приложения часто используют этот тип передачи данных.

Для своей работы я решил использовать протокол TCP/IP так как он более надежен по сравнению UDP и обеспечивает целостность данных при отправке.

1. **Разработка интерфейса взаимодействия пользователя с системой**

Для удобного взаимодействия пользователя был разработан следующий интерфейс.

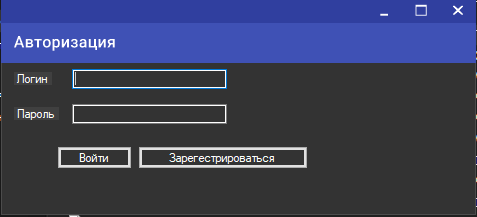


Рисунок 3.4 – Окно авторизации

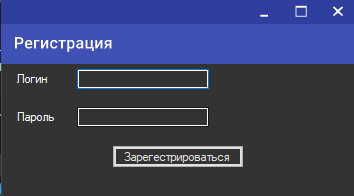


Рисунок 3.5 – Окно регистрации



Рисунок 3.6 – Просмотр чатов

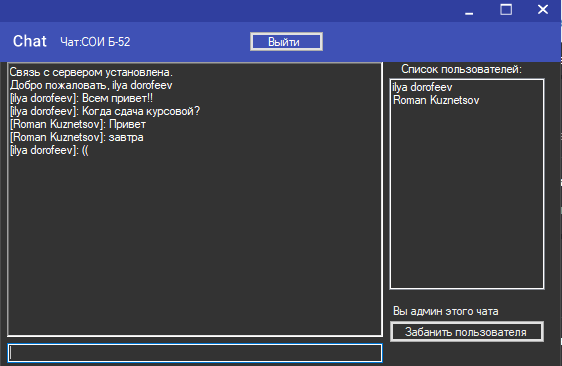


Рисунок 3.7 – Чат

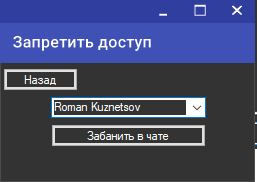


Рисунок 3.8 – Блокировка пользователя

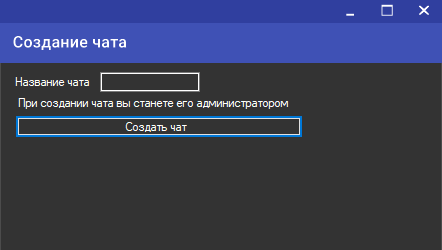


Рисунок 3.9 – Создание чата

# 4. ПРОЕКТНО – ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

**4.1. Тестирование и отладка макета рабочей программы.**

Тестирование приложения. Для начала необходимо запустить веб-приложение чата. После запуска приложения, открывается окно с авторизацией(рисунок 4.1).

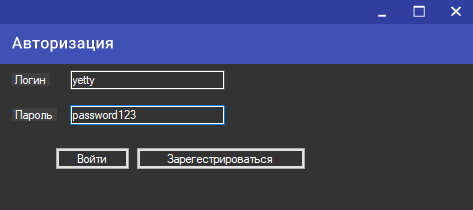
****

Рисунок 4.1 – Пример авторизации

Одной из самых первоначальных и главных задач веб-приложения является авторизация и регистрация пользователей. Для этого необходимо проверить правильность введенных данных. Для этого приведем пример ввода с ошибкой и проверим корректность работы приложения (рис. 4.2).

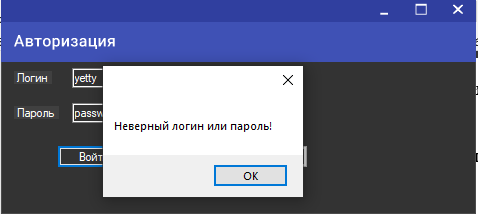


Рисунок 4.2 – Обработка несуществующего логина или пароля

При регистрации пользователя не допускается повтор логина. Для проверки введем логин который уже занят.



Рисунок 4.3 – Обработка существующего логина

Таким образом, было выяснено, что веб-приложение выдает предупреждение о том, что пользователя с таким логином уже существует.

Попробуем создать чат с посощью кпопки создания чата и проверить появился ли он в списке чатов(рис. 4.4,4.5,4.6).

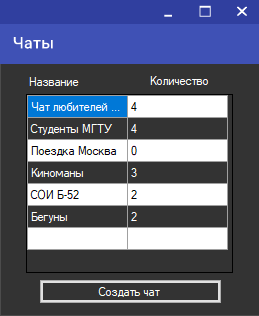


Рисунок 4.4 – Окно чатов

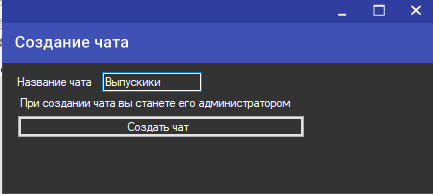


Рисунок 4.5 – Окно создания чата

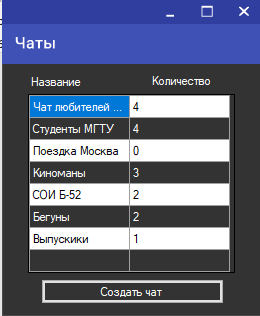


Рисунок 4.6 – Окно чатов после создания чата

Можно увидеть, что после создания нового чата, он появляется в окне с другими чатами и в параметр количества участников равняется одному, так как создатель чата является его администратором и единственным участником.

Откроем еще один клиент чата и залогинемся в уже созданный чат(рис. 4.7).

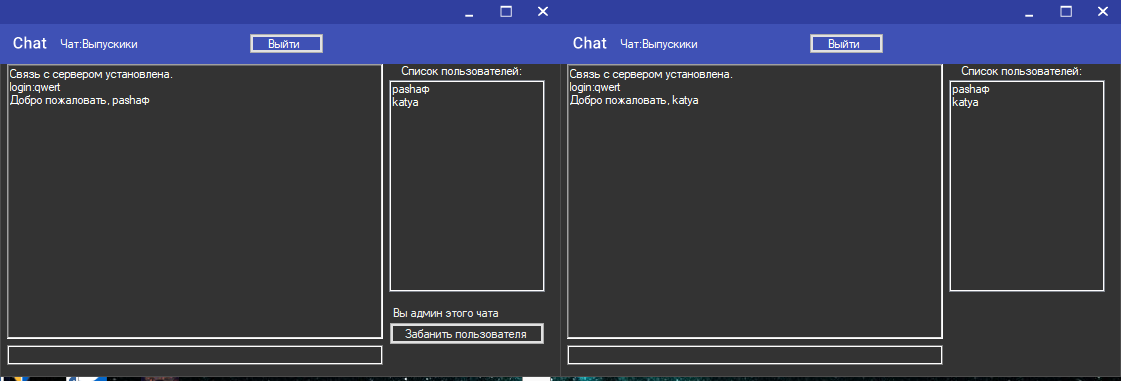


Рисунок 4.7 – Окно чата

После поткрытия двух чатов мы видим что в чате администратора присутствует кнопка блокировки пользователей, а в обычном чате такой кнопки нет.

Проверим отправку сообщений из одного клиента в другой(рис. 4.8).

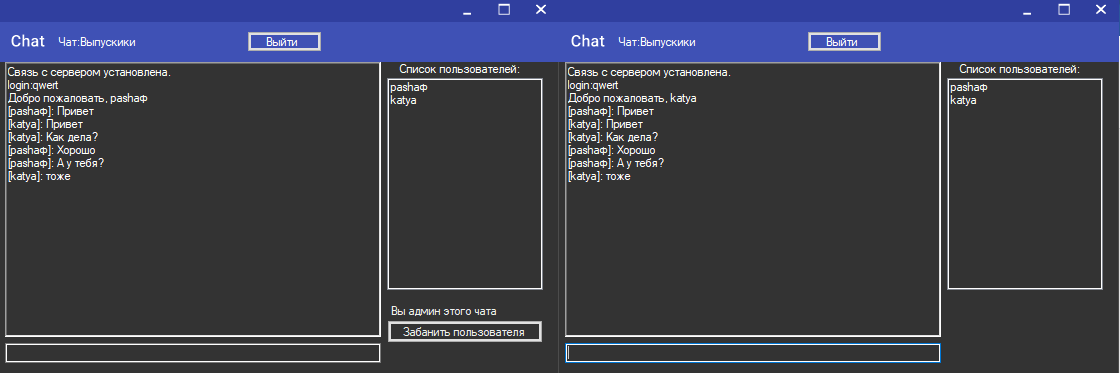


Рисунок 4.8 – Окно чата после отправки сообщений

Как мы видим сообщения из одного клиента отправляются на другой.

Проверим механизм блокировки пользователей. Для этого нажмем на кнопку «Заблокировать пользователя» и выберем пользователя которого хотим заблокировать. (рис. 4.8 и 4.9).

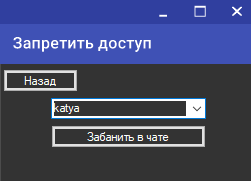


Рисунок 4.8 – Блокировка пользователя

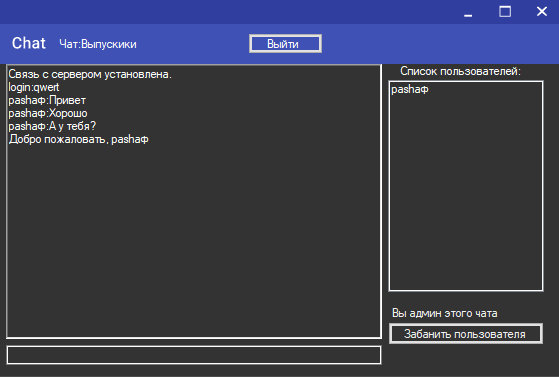


Рисунок 4.9 – Удаление сообщений заблокированного пользователя

Как мы видим после блокировки пользователя его сообщения удалились и сессия была завершена.

**4.2. Разработка руководства администратора**

Для администрирования серверной части приложения требуется распаковать файл .exe c сервером на постоянно запущенном компьютере или воспользоваться хостингом серверов, где разместить код сервера.

Шаг 1 Распаковка файла

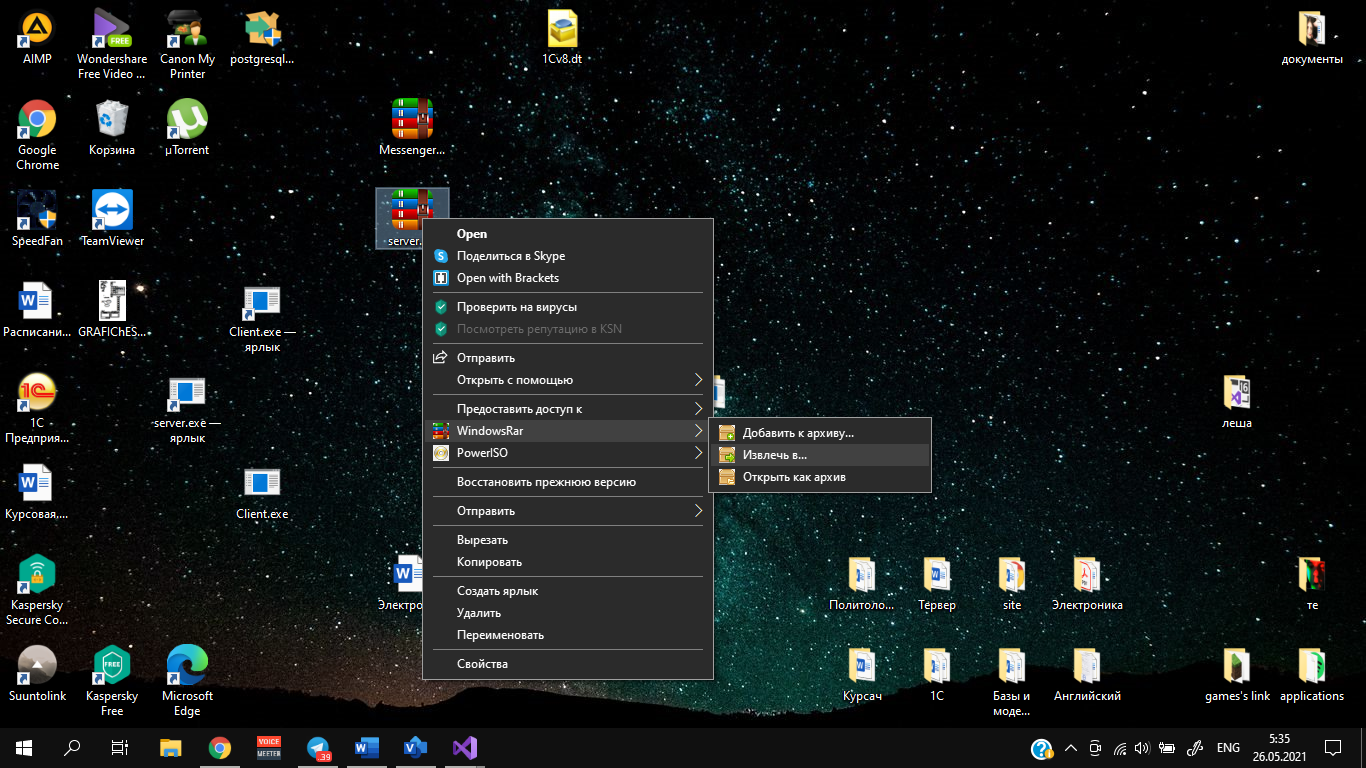


Рисунок 4.10 – Распаковка файла

Шаг 2 Заупск приложения

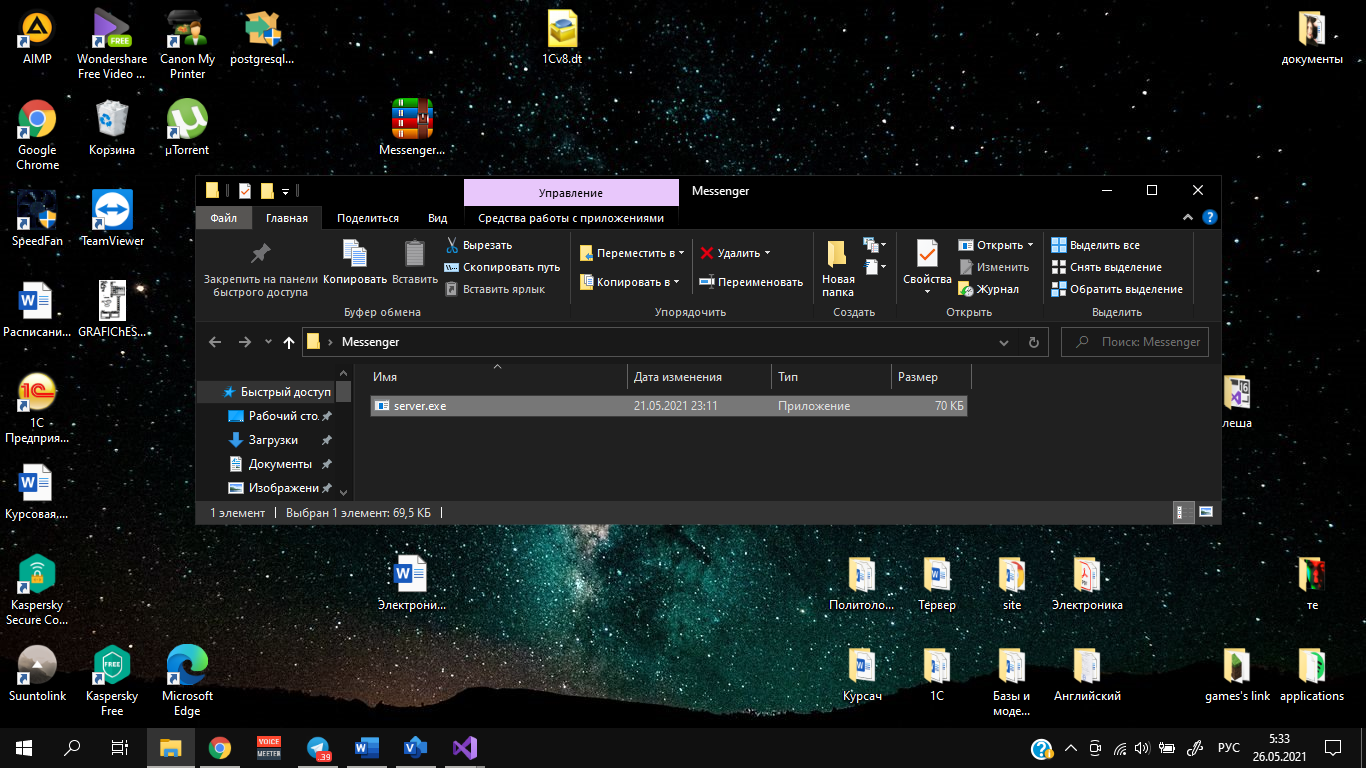
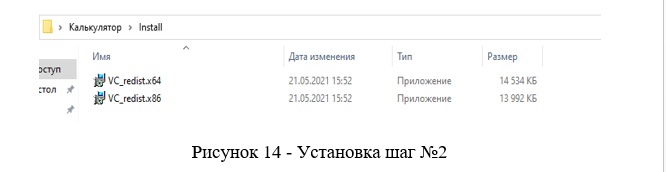
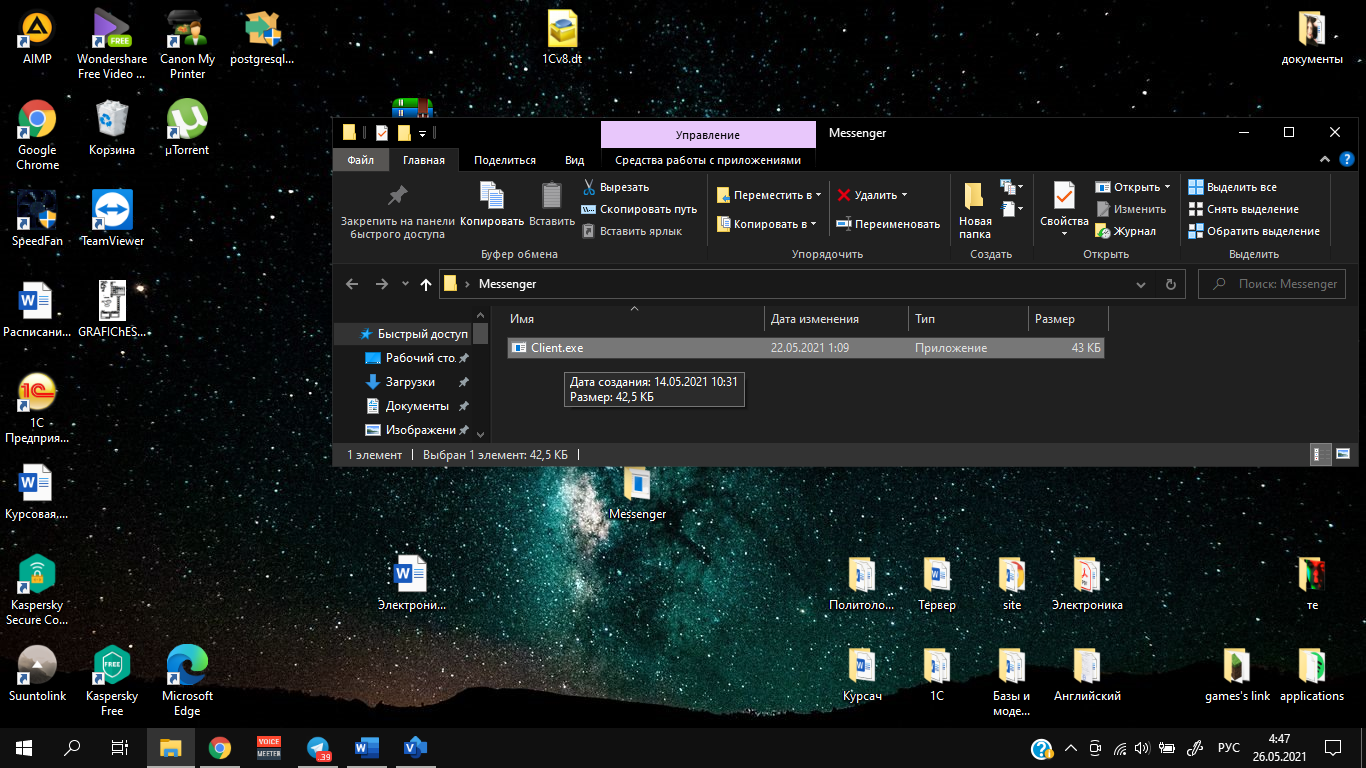
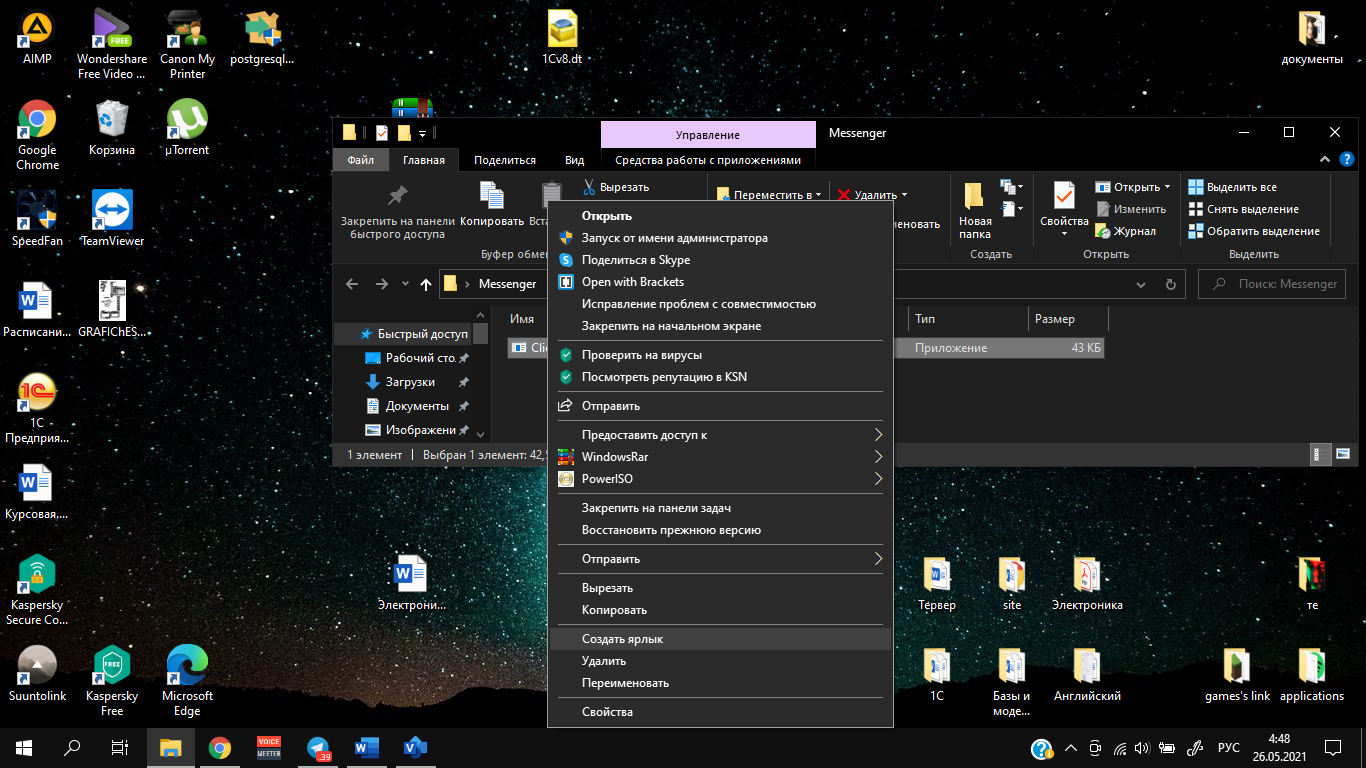


Рисунок 4.10 – Приложение сервера

**4.3. Разработка руководства пользователя**

Для того, чтобы пользоваться программой, необходимо сделать распаковку архива с приложением (рисунок 13).  
  
Рисунок 4.11 – Установка шаг №1

Если вы до этого ни разу не пользовались Visual Studio, то вам следует открыть папку «Install», которая находится внутри основной папки «Чат» и установить необходимые драйвера для успешной работы с приложением (Рисунок 4.12 ).  
  
 Рисунок 4.12– Установка шаг №2  
Приложение готово к работе! Все что осталось сделать – это открыть исполняемый файл с расширением .exe.  
  
Рисунок 4.13 – Установка шаг №3

Если вы планируете пользоваться чатом на постоянной основе, то вам будет удобнее создать ярлык на рабочий стол, чтобы быстрее производить запуск приложения. Для этого необходимо нажать правой кнопкой мыши по исполняемому фалу .exe, после этого выбрать пункт «Создать ярлык» (Рисунок 4.14).  
  
Рисунок 4.10 – Создание ярлыка калькулятора  
  
После этого можно перенести ярлык на рабочий стол, и при двойном нажатии на него, будет открываться приложение.

**4.4. Проектирование начального и тестового наполнения базы данных. Порядок развертывания системы.**

Содержимое базы данных является основным источником информации для данного приложения.

Для того чтобы развернуть базу данных в запроснике PosgresSQL необходимо ввсести:

CREATE TABLE Messages(

id\_mess serial PRIMARY KEY,

message TEXT,

id\_user INTEGER ,

id\_chat INTEGER ,

FOREIGN KEY (id\_user) REFERENCES users (id),

FOREIGN KEY (id\_chat) REFERENCES chats (id));

CREATE TABLE chats(

id serial PRIMARY KEY,

chat\_name TEXT,

number\_of\_members INTEGER ,

admin INTEGER);

CREATE TABLE users(

id serial PRIMARY KEY,

login TEXT,

password TEXT);

CREATE TABLE bans(

id\_user INTEGER ,

id\_chat INTEGER ,

FOREIGN KEY (id\_user) REFERENCES users (id),

FOREIGN KEY (id\_chat) REFERENCES chats (id));

# 5. ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы реализован многопользовательский чат с возмножностью авторизации, выбора переписок а также блокировкой пользователей и хранения всей информации в базе данных.

В процессе выполнения курсовой работы приобретен опыт в оконных приложений в С#.

Изучена предметная область, проведен анализ аналогов и прототипов, осуществлен выбор инструментов и платформы для разработки. Реализована структура базы данных, написана серверная часть приложения и разработан пользовательский интерфейс.

В данной работе выполнены все поставленные задачи.

# 6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТУРАТУРЫ

1. Крис Касперски «Самоучитель игры на WINSOCK» 2004 г.,22 стр.
2. Джонс,Оланд  «Программирование в сетях Microsoft Windows» 2004 г., 321 стр.
3. Работа с базами данных на языке С#. Технология ADO.net. Учебное пособие. Ульяновск 2009
4. Документация PostgreSQL [Электронный ресурс] – URL: <https://postgrespro.ru/docs>
5. Многопоточность [Электронный ресурс] – URL: <https://metanit.com/sharp/tutorial/11.1.php>
6. Джеффри Рихтер «CLR via C#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C#». 2017 г., 896 стр.
7. Билл Вагнер «Наиболее эффективное программирование на C#. 50 способов улучшения кода». 2017 г., 240 стр.