МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема «Мессенджер Katran»

Исполнитель

студент 2 курса 5 группы Гунько Дмитрий Витальевич

Руководитель работы ассистент Северинчик Н.А

Курсовой проект защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Председатель Пацей Н.В.

Минск 2020

# **Содержание**

[**Содержание** 2](#_Toc41278674)

[**Введение** 2](#_Toc41278675)

[**1. Аналитический обзор литературы** 3](#_Toc41278676)

[**2.** **Анализ требований к программному средству и разработка функциональных требований** 6](#_Toc41278677)

[2.1. Описание инструментов разработки программного средства 6](#_Toc41278678)

[2.2. Описание функций, реализуемых программой 8](#_Toc41278679)

[**3.** **Проектирование программного средства** 10](#_Toc41278680)

[3.1 Проектирование логической структуры базы данных 10](#_Toc41278681)

[3.2 Диаграмма последовательности 11](#_Toc41278682)

[3.3 Диаграмма действий 11](#_Toc41278683)

[**4.** **Создание (реализация) программного средства** 12](#_Toc41278684)

[4.1 Реализация библиотеки классов 12](#_Toc41278685)

[4.2 Реализация серверной части 14](#_Toc41278686)

[4.3 Реализация клиентской части приложения 16](#_Toc41278687)

[4.3.1 Обобщенная структура 16](#_Toc41278688)

[4.3.2 Разбор реализации клиентской части 18](#_Toc41278689)

[**5.** **Тестирование программного средства** 20](#_Toc41278690)

[**6.** **Руководство пользователя** 22](#_Toc41278691)

[**ЗАКЛЮЧЕНИЕ** 29](#_Toc41278692)

[**Список использованных источников** 30](#_Toc41278693)

# **Введение**

Мессенджер – это программа на ПК, позволяющая мгновенно обмениваться сообщениями между пользователями. Для подобного рода коммуникации необходима клиентская программа, так называемый мессенджер. Отличие от электронной почты здесь в том, что обмен сообщениями идёт в реальном времени. Большинство IM-клиентов позволяет видеть, подключены ли в данный момент абоненты, занесённые в список контактов. В ранних версиях программ всё, что печатал пользователь, тут же передавалось. Если он делал ошибку и исправлял её, это тоже было видно. В таком режиме общение напоминало телефонный разговор.

В современных программах сообщения появляются на мониторе собеседника уже после окончания редактирования и отправки сообщения. Как правило, мессенджеры не работают самостоятельно, а подключаются к центральному компьютеру сети обмена сообщениями, называемому сервером. Поэтому мессенджеры и называют клиентами (клиентскими программами). Термин является понятием из клиент-серверных технологий. Широкому кругу пользователей известно некоторое количество популярных сетей обмена сообщениями, таких как IRC, Skype, AIM, ICQ, MSN, Yahoo!, Jitsi , XMPP. Каждая из этих сетей разработана отдельной группой разработчиков, имеет отдельные серверы и протоколы, отличается своими правилами и особенностями. Между различными сетями обычно нет прямой связи (только в XMPP существует понятие межсетевого транспорта), таким образом, пользователь сети Skype не может связаться с пользователем сети ICQ, однако ничто не мешает быть одновременно пользователем нескольких сетей.

Цель моего курсового проекта разработать мессенджер для обмена сообщениями между пользователями с возможностью добавления в контакты, созданием бесед и обменом как сообщениями так и файлами.

Технологии, используемые в работе:

1. Объектно-ориентированный язык программирования C#;
2. Система для построения клиентских приложений WPF;
3. Технология для работы с базами данных ADO.NET;
4. Шаблон проектирования MVVM;
5. Среда разработки Visual Studio 2019;
6. Система управления реляционными базами данных Microsoft SQL Server;
7. Язык T-SQL для написания запросов к БД;
8. Клиент-серверное взаимодействие по протоколу TCP.

# **1. Аналитический обзор литературы**

Telegram — программа, которая призвана заменить обмен привычными сообщениями. SMS, Hangouts, Viber, WhatsApp, может, даже Gmail. Ведь пересылать можно не только текст, но и файлы разных форматов — от картинок до документов. Приложение привязывается к номеру телефона, приходит проверочное сообщение с кодом (на мобильной версии для Android код подтягивается автоматически). Общаться можно как с конкретными пользователями, которые уже установили приложение, так и организовывать групповые чаты до 200 человек и отправлять пригласительные сообщения с призывом установить себе эту крутотенюшку. Еще раз — двести человек в чате! Разработчики считают это одной из самых главных фишек своего приложения (рис 1.1 ).

Благодаря защищенности мессенджера и защите от перехвата приложение стало популярно в Гонконге во время протестов. Изначально о митинге договаривались через Facebook, но эту информацию легко заполучить.

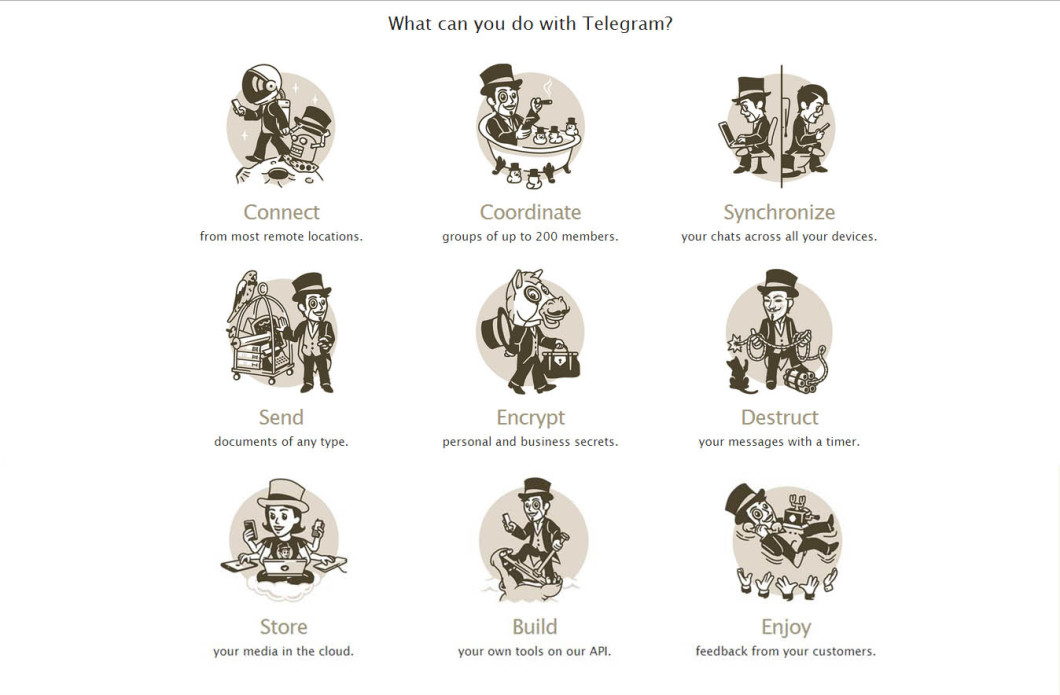


Рисунок 1.1 – Возможности Мессенджера Telegram

Viber — приложение-мессенджер, которое позволяет отправлять сообщения, совершать видео- и голосовые VoIP-звонки через интернет. Голосовые вызовы между пользователями с установленным Viber бесплатны. Также в Viber имеется возможность передачи изображений, видео- и аудиосообщений, документов и файлов.

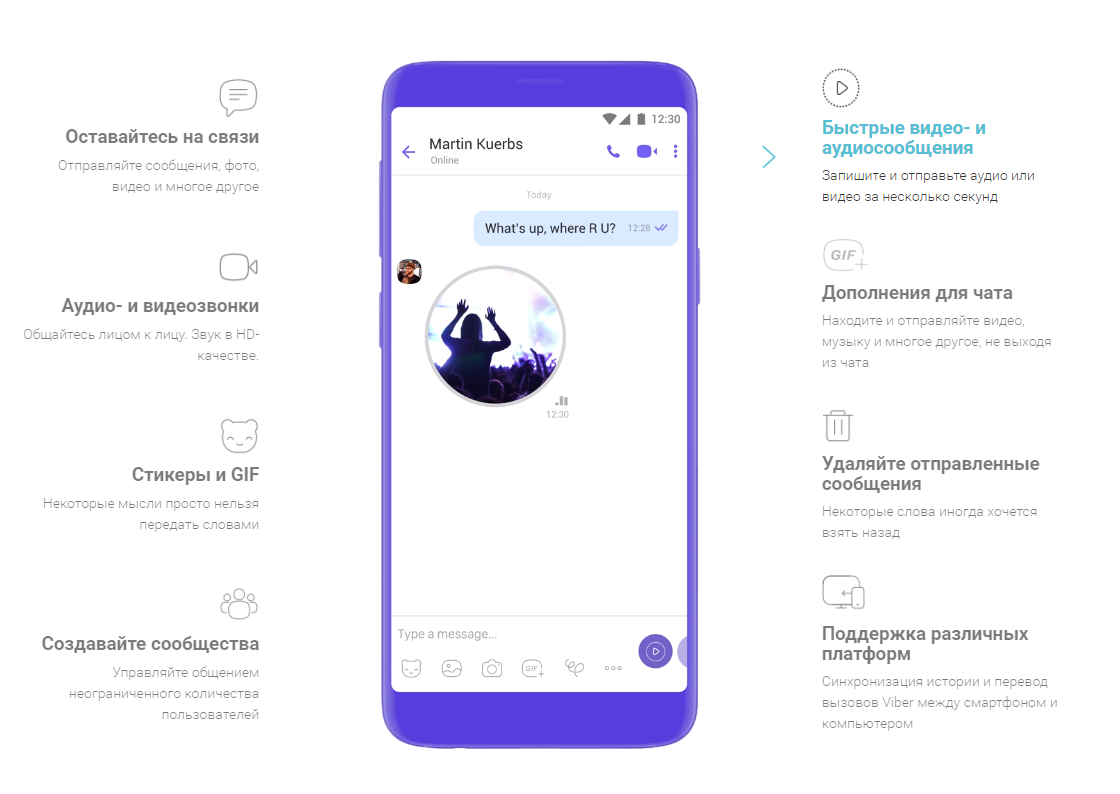


Рисунок 1.2 – Возможности Мессенджера Viber

ICQ (акроним от англ. I seek You/"Я ищу тебя") – популярный мессенджер для мгновенного обмена текстовыми сообщениями и анимационными изображениями, ныне находящийся в собственности российской группы Mail.ru.

Достоинства

1. красивый пользовательский интерфейс (привлекательная цветовая гамма, новые смайлы, радующая взор анимация и т.д.);
2. исключительная простота пользования;
3. возможность общения одновременно с несколькими собеседниками (с опцией параллельного просмотра уведомлений в разных окнах);
4. простор для творчества в контексте "подгонки" оформления под свой вкус (замена иконок, изменение "обоев", опция добавления смайликов от сторонних разработчиков и пр.);
5. опция группирования контактов (объединение собеседников в группы);
6. возможность установки на различные девайсы (от настольных ПК до смартфонов);
7. опция отправки файлов;
8. возможность "привязки" программы к соцсетям (позволяет отыскать потерявшихся одноклассников, коллег, сослуживцев и т.д. по их сетевым аккаунтам);
9. опция проведения групповых видеоконференций;
10. малая нагрузка программы на компьютер;
11. быстрая закачка и установка.

Недостатки

1. надоедливые окна с рекламой;
2. периодическая "потеря" сообщений;
3. на самая надежная защита от спама;
4. отсутствие истории переписки (в отдельных версиях).

# **Анализ требований к программному средству и разработка функциональных требований**

## 2.1. Описание инструментов разработки программного средства

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2019;
* программная платформа .NET Framework 4.7.2;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* ADO.NET;
* Material Design Library;
* Протокол TCP;

*C#* — объектно-ориентированный язык программирования. Разработан в 1998 — 2001 под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework.

XAML — это декларативный язык разметки. С точки зрения модели программирования .NET Framework язык XAML упрощает создание пользовательского интерфейса для приложения .NET Framework.

WPF (Windows Presentation Foundation) — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

ADO.NET — предоставляет собой технологию работы с данными, которая основана на платформе .NET Framework. Эта технология представляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

Причем важно отметить, что систем управления баз данных может быть множество. В своей сущности они могут различаться. MS SQL Server, например, для создания запросов использует язык T-SQL, а MySQL и Oracle применяют язык PL-SQL. Разные системы баз данных могут иметь разные типы данных. Также могут различаться какие-то другие моменты. Однако функционал ADO.NET построен таким образом, чтобы предоставить разработчикам унифицированный интерфейс для работы с самыми различными СУБД.

Material Design Library — это исчерпывающая кросс-платформенная концепция визуального оформления приложений, затрагивающая правила отображения анимации, общий стиль приложения и правила взаимодействия с управляющими элементами.

TCP-протокол — огромное количество трафика и взаимодействия в сети сейчас происходит по протоколу TCP (Transmission Control Protocol). Этот протокол гарантирует доставку сообщений и широко используется в различных существующих на сегодняшний день программах.

Для работы с протоколом TCP в .NET предназначены классы TcpClient и TcpListener. Эти классы строятся поверх класса System.Net.Sockets.Socket. На основе этого протокола реализовано общение клиент-серверное общение.

## 2.2. Описание функций, реализуемых программой

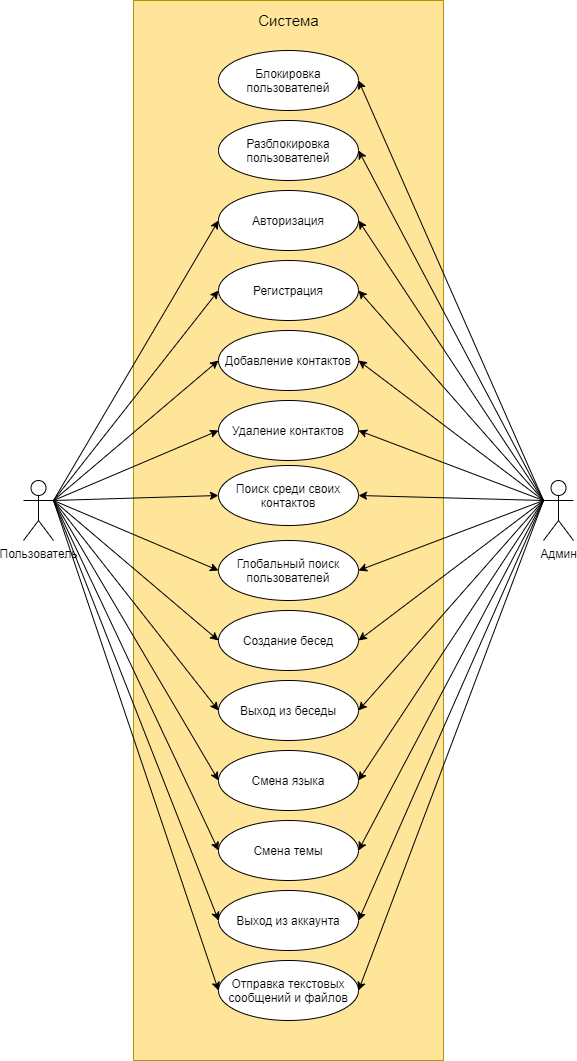


Рисунок 2.1 – Диаграмма вариантов использования (UML Case)

Рассмотрим каждый из прецендентов поподробнее.

Прецендент «Блокировка пользователей» заключается в блокировке выбранного пользователя из списка навсегда.

Прецендент «Разблокировка пользователей» заключается в разблокировке выбранного пользователя из списка.

Прецендент «Авторизация» заключается во входе в приложение с данными уже существующего пользователя.

Прецендент «Регистрация» заключается в создании нового пользователя с указанием его логина, имени в приложении, пароля, почты и аватарки(по желанию). Доступно только создание пользователей с правами доступа «Пользователь».

Прецендент «Добавление контактов» заключается в добавлении в свою сеть контактов пользователя с создание чата с ним.

Прецендент «Удаление контактов» заключается в удалении пользователя из личной сети контактов вместе с чатом.

Прецендент «Поиск среди своих контактов» заключается в фильтрации сети своих контактов по имени.

Прецендент «Глобальный поиск пользователей» заключается в поиске пользователей по имени вне своих контактов.

Прецендент «Создание бесед» заключается в создании новой беседы с указанием ее названия, аватарки(по желанию) и выбором участников из списка своих контактов.

Прецендент «Выход из беседы» заключается в удалении из сети своих контактов беседы, если участников больше не осталось, то удаляется и сама беседа.

Прецендент «Смена языка» заключается в смене языка приложения на 1 из доступных и сохранение этого в настройках приложения, при следующем входе в приложение будут применены те настройки, которые были установлены в прошлый раз.

Прецендент «Смена темы» заключается в смене темы приложения на 1 из доступных и сохранение этого в настройках приложения, при следующем входе в приложение будут применены те настройки, которые были установлены в прошлый раз.

Прецендент «Выход из аккаунта» заключается в выходе из аккаунта. При последующем включении приложения потребуется заново ввести данные для входа или зарегистрироваться.

Прецендент «Отправка текстовых сообщений и файлов» заключается в обмене пользователями сообщениями, которые могут содержать как файлы, так и текст.

# **Проектирование программного средства**

## Проектирование логической структуры базы данных

Для реализации поставленной задачи была создана база данных Katran. Для её создания использовалась система управления реляционными базами данных Microsoft SQL Server. В качестве IDE: DataGrip 2019.3.3.

Прежде всего, необходимо спроектировать корректную базу данных для работы. В моем случае, база данных должна содержать следующие таблицы:

* Данные для авторизации;
* Полная информация о пользователях;
* Контакты пользователей;
* Информация о чатах;
* Участники чатов;
* Сообщения чатов (создаются динамически для каждого чата отдельно).

На рисунке 3.1 представлена диаграмма базы данных

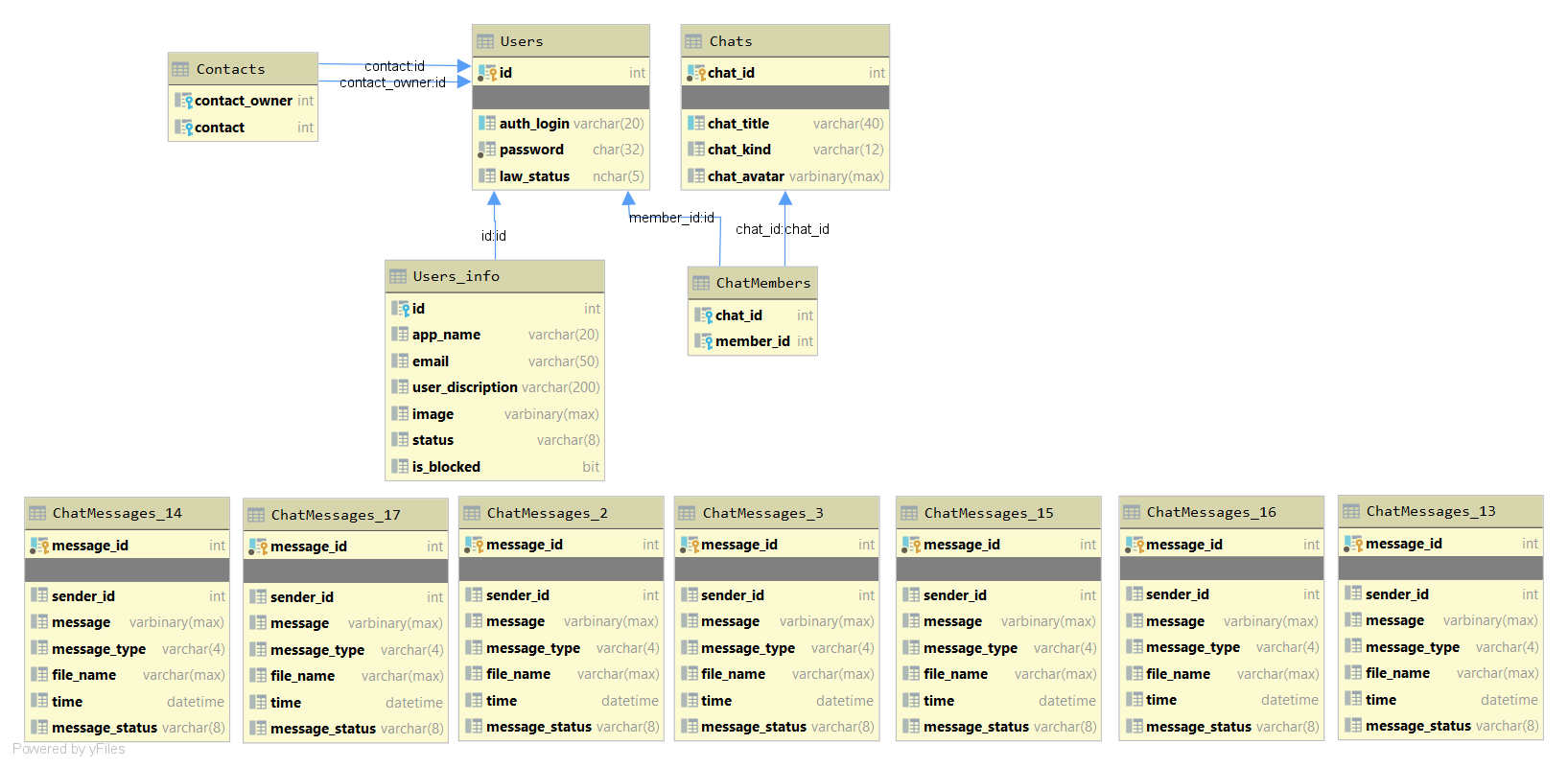


Рисунок 3.1 – Диаграмма базы данных

Основными таблицами выступают таблицы Users и Chats – они содержат информацию для авторизации и основную информацию о чатах соответственно. Таблицы ChatMessages\_[номер чата] создаются динамически при создании чата или беседы и удаляются соответственно при их удалении.

## Диаграмма последовательности

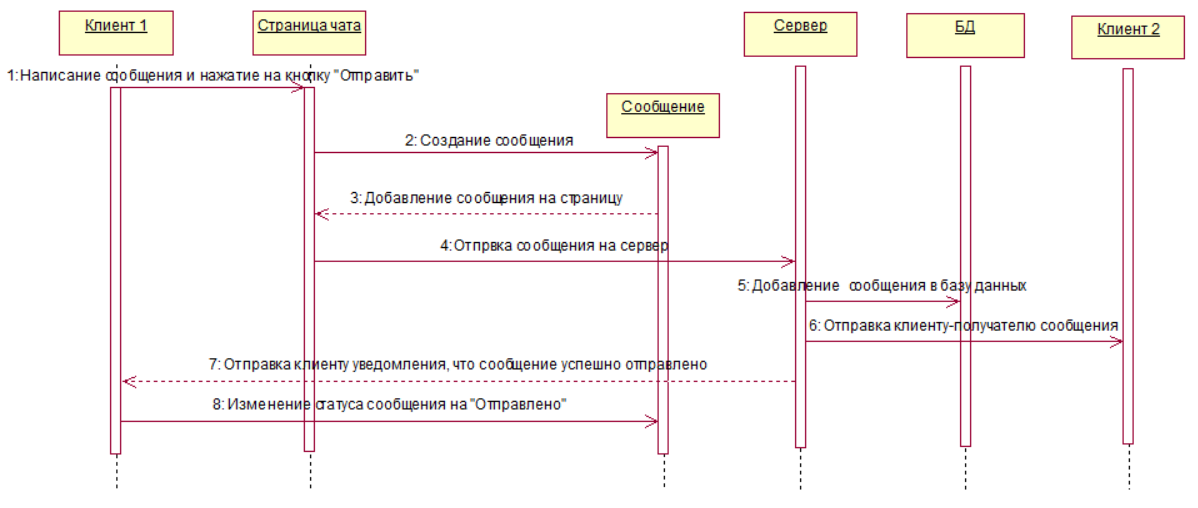


Рисунок 3.2 – Диаграмма последовательности при отправке сообщения

На данной диаграмме показана последовательность действий, происходящая при отправке сообщения другому пользователю. Чтобы отправить сообщение клиент 1 обращается к странице, чтобы та создала на основе введеного текста сообщение. После создания сообщения оно отправляется на страницу клиенту 1 со статусом «Не отправлено». Далее со страницы клиента 1 отправляется запрос на сервер. Сервер записывает только что пришедшее сообщение в базу данных и отправляет его клиенту-получателю. Если все прошло успешно, то клиент получает ответ от сервера, что сообщение отправлено и меняет его статус на «Отправлено».

## Диаграмма действий

Данная диаграмма отображает процесс деятельности пользователя в процессе работы с приложением.

Диаграмма деятельности позволяет любому, кто выполняет данный процесс, выбирать порядок действий. Другими словами, диаграмма только устанавливает правила обязательной последовательности действий, которым пользователь должен следовать. Это важно для моделирования бизнес-процессов, поскольку эти процессы часто выполняются параллельно.

На рисунке 3.3 представлена диаграмма деятельности мессенджера «Katran».

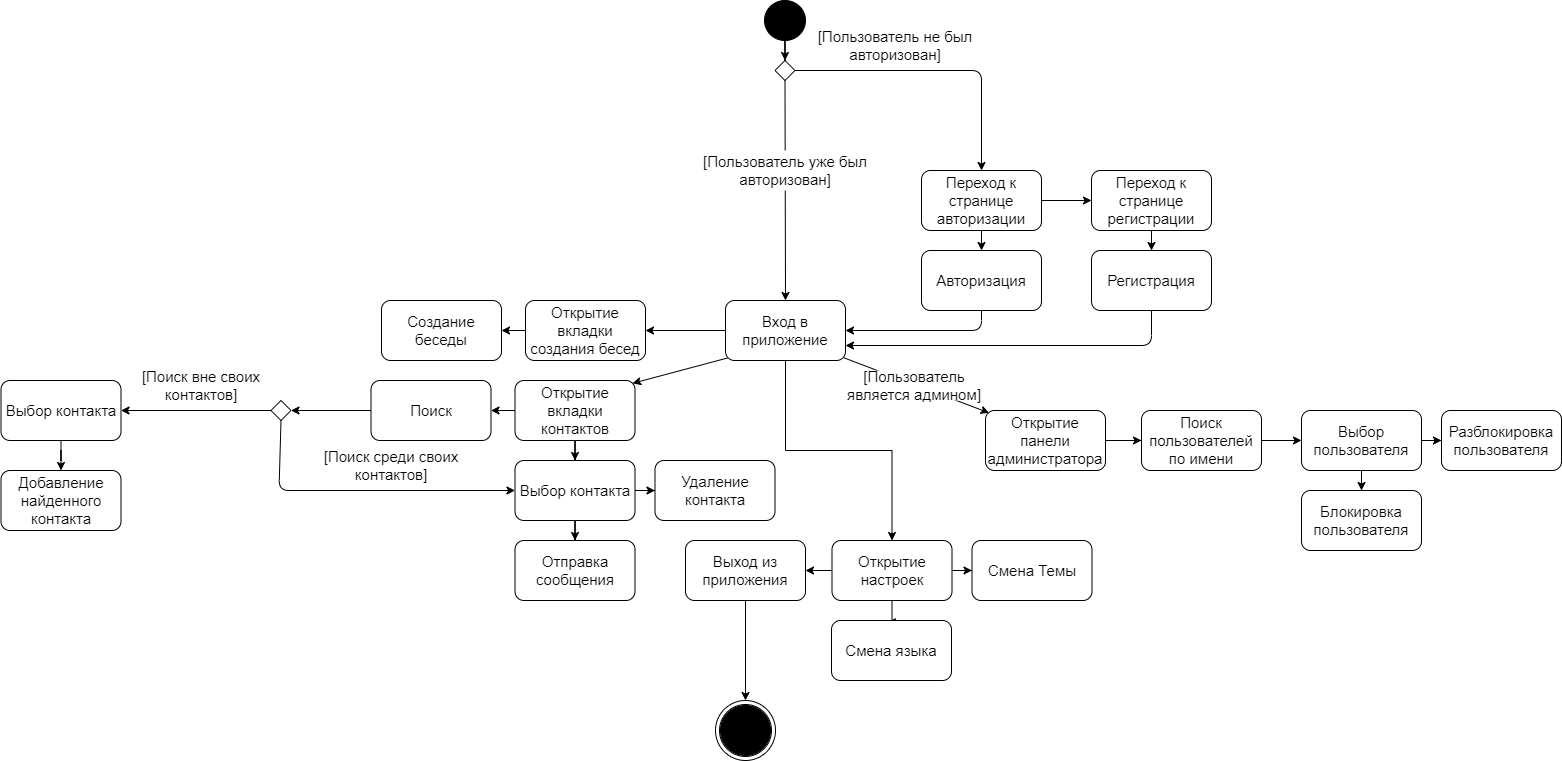


Рисунок 3.3 – Диаграмма деятельности

# **Создание (реализация) программного средства**

### Реализация библиотеки классов

Библиотека классов представляет собой описание классов, на основе которых происходит клиент-серверное взаимодействие. Диаграмма классов изображена на рисунках 4.1 и 4.2.

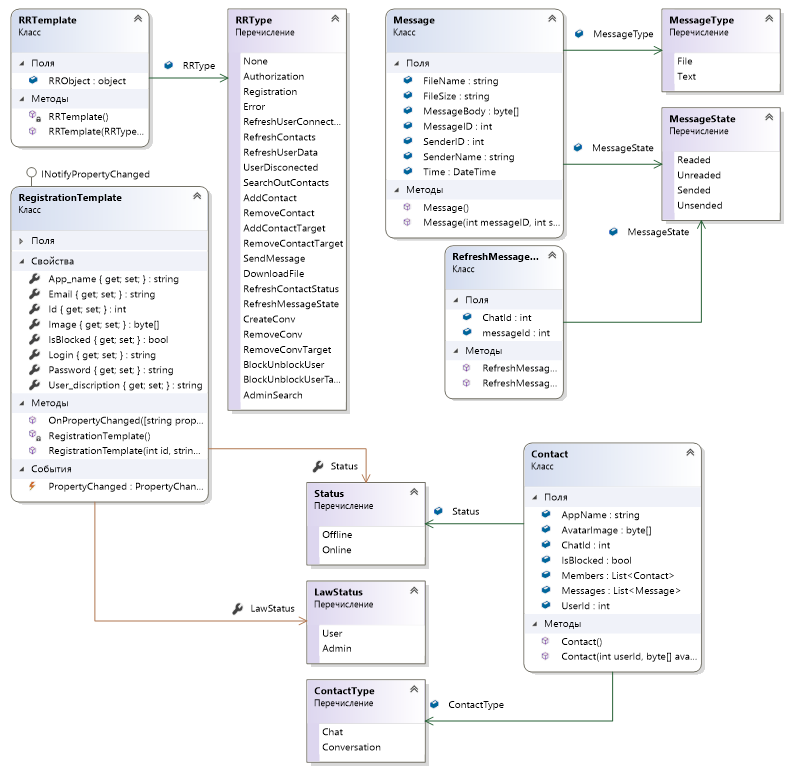


Рисунок 4.1 – Диаграмма классов библиотеки

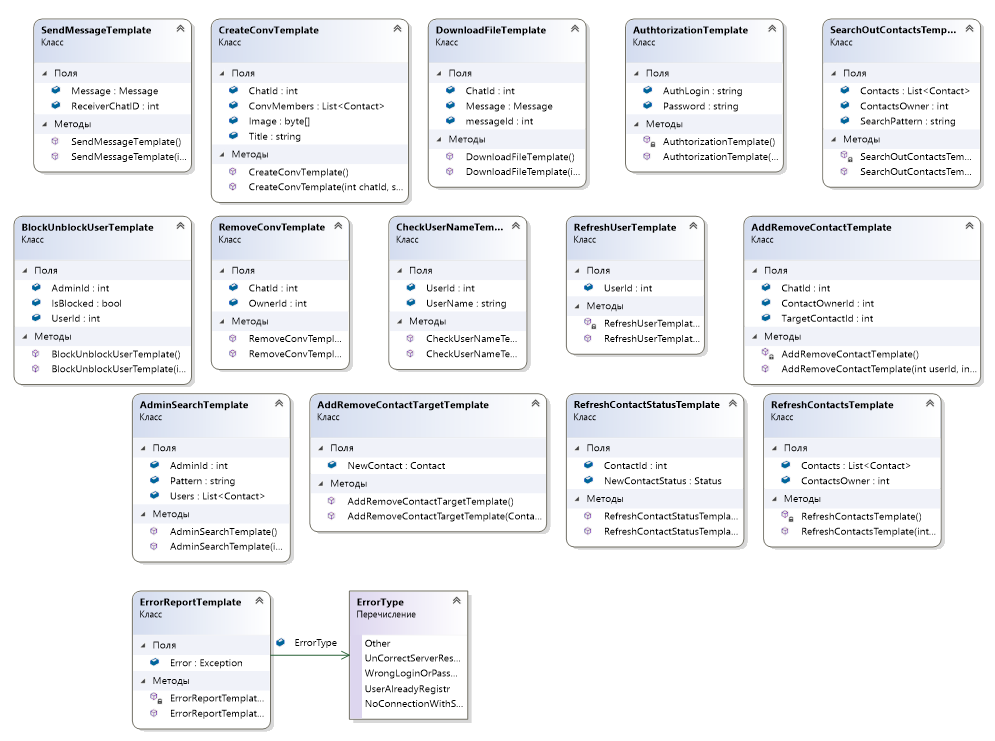


Рисунок 4.2 – Диаграмма классов библиотеки

Главным классом во взаимодействии выступает класс RRTemplate, диаграмма которого изображена на рисунке 4.3. В поле RRType хранится значение перечисления, которое сигнализирует о типе объекта, упакованного в RRObject.

Благодаря этому классу сервер и клиент могут взаимодействовать через одну сущность. Что является очень гибким решением и позволяет горизонтально расширять функционал.

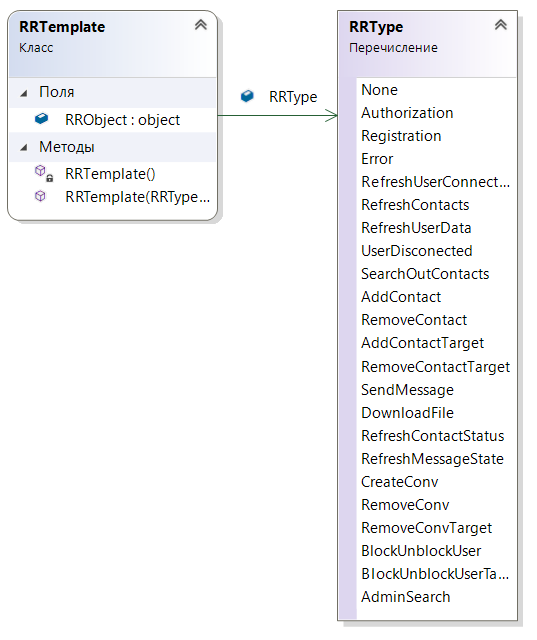


Рисунок 4.3 – Диаграмма класса RRTemplate

## Реализация серверной части

В роли сервера выступает консольное приложение .NET Framework.

Сервер реализован на основе TCP-протокола, в связи с его надежностью передачи данных. Сервер является многопоточным, при каждом запросе с клиентской части на сервер создается новый поток, в котором запрос обрабатывается и возвращается ответ. На рисунке 4.4 приведен пример кода, где реализовано создание потока под каждый запрос с клиентской части.

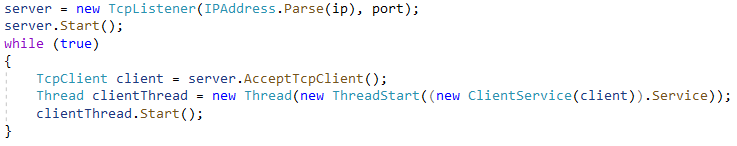


Рисунок 4.4 – Реализация многопоточного сервера

Функция AcceptTcpClient возвращает TCP-оболочку сокета пользователя. Пока на сервер не поступает никаких запросов он находится в режиме ожидания.

В функцие Service объекта ClientService происходит обработка запроса клиента и отправка соответствующего ответа.

На рисунке 4.5 представлен пример кода получения данных из пользовательского потока и приведения их к нужному типу данных посрудством поля RRType в объекте типа данных RRTemplate, который служит классом-обменником.



Рисунок 4.5 – Листинг функции Service

Подключение к базе данных происходит посредством строки подключения и и технологии ADO.NET, для контроля единственности подключения к бд реализован паттерн Singleton. Диаграмма классов серверной части представлена на рисунке 4.6.

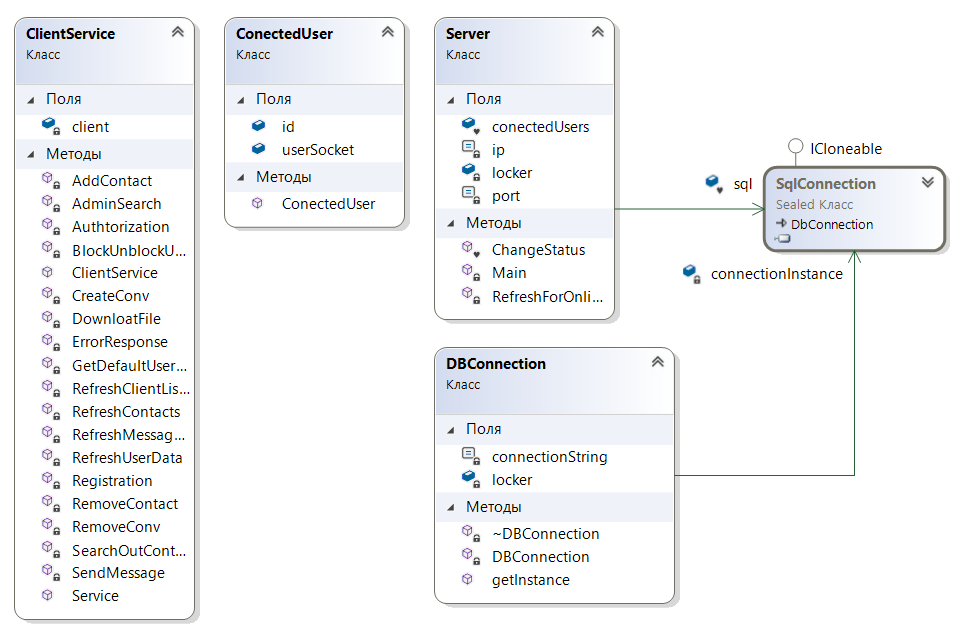


Рисунок 4.6 – Диаграмма классов серверной части

## Реализация клиентской части приложения

### Обобщенная структура

При создании приложеия были реализованны такие паттерны как MVVM и Command.

Структура клиентской части приложения представлена на рисунке 4.7.

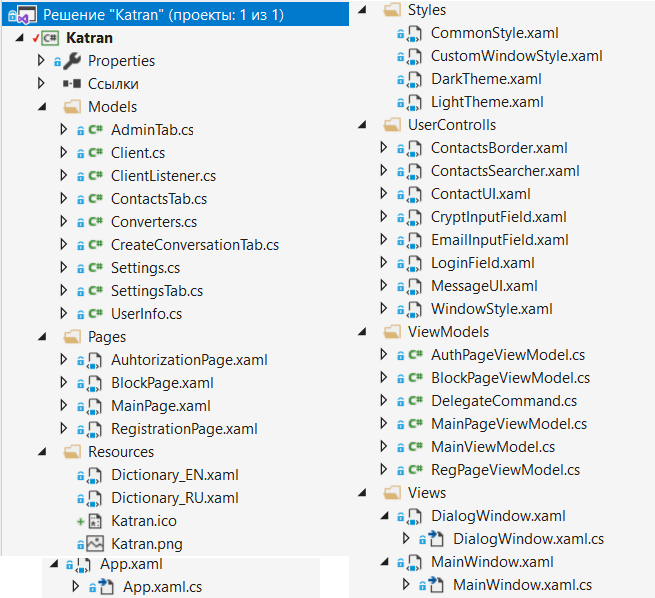


Рисунок 4.7 – Структура клиентской части приложения

Как показано на рисунке, все элементы проекта, согласно паттерну MVVM, рассредоточены в специальных папках для создания целостности и упрощения организованности приложения:

* Models - модель данных, описывает используемые в приложении данные. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления;
* Pages – основные страницы окна приложения;
* Resources и Styles - содержат в себе словари ресурсов, необходимые для задания стилей для элементов управления и установления языка приложения;
* UserControlls – собственные элементы управления для предотвращения дублирования кода и сохранения единообразия стилей;
* ViewModels - модель представления, связывает модель и представление через механизм привязки данных;
* View - содержит в себе все представления, которое определяет визуальный интерфейс, через который пользователь взаимодействует с приложением.

## Разбор реализации клиентской части

Диаграмма классов клиентской части приложения представлена на рисунке 4.8.

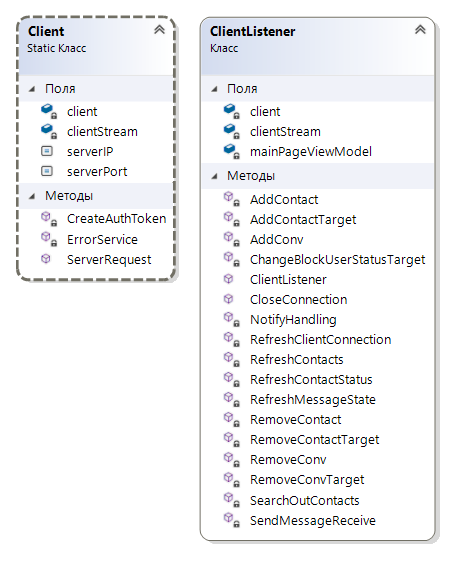


Рисунок 4.8 – Диаграмма классов клиентской части приложения

Класс Client – является статическим классом, с помощью которого отправляются единоразовые запросы на сервер. Для каждого нового запроса в нем создается новое подключение и после того, как ответ от сервера поступил, если он ожидается, то подключение закрывается.

Класс ClientListener – класс, который выступает в роле слушателя. Начинает свое существование как только пользователь авторизуется и попадает на главную страницу. В этот момент он отправляет на сервер уведомление, что пользователь появился в сети. Как только сервер принимает этот запрос он записывает подключенный сокет в список и не закрывает с ним соединение, чтобы можно было отправлять сообщения. Данный класс так же имеет доступ к MainPageViewModel, что дает ему возможность взаимодействовать с приложением: отображать только что добавленные контакты, пришедшие сообщения и т.д. без перезагрузки приложения.

На рисунке 4.9 представлены общая картина взаимодействия классов

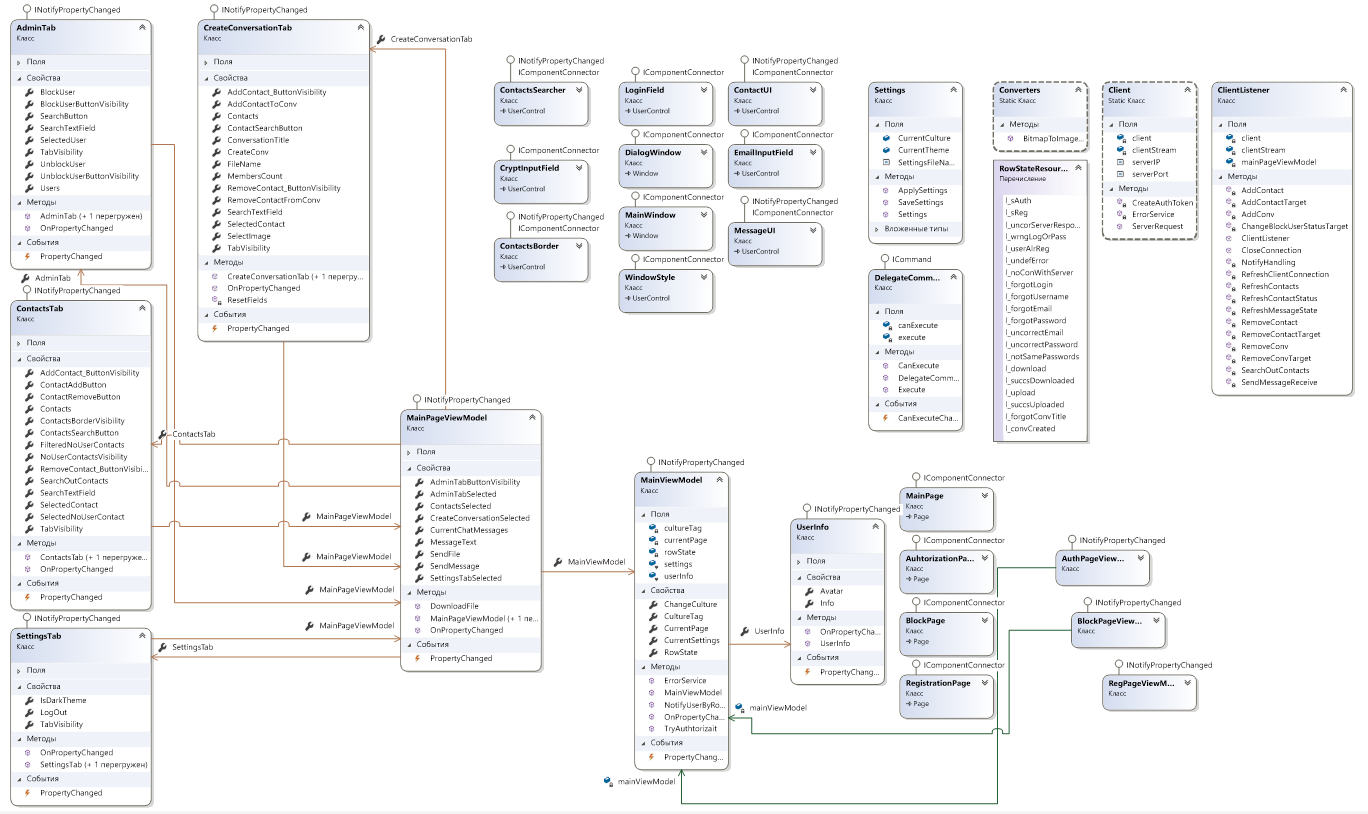


Рисунок 4.9 – Общая картина взаимодействия классов

# **Тестирование программного средства**

При разработке приложение было сконструировано так, чтобы все данные, с которыми работает пользователь вводились корректно. Это относится к форме регистрации и входа.

На рисунке 5.1 продемонстрировано поведение страницы авторизации при отсутствии записей в полях. Такое же предупреждение будет возникать и в случае неправильного ввода пароля или логина. Так же в левом нижнем углу присутствует появляющаяся строка состояния, которая уведомляет пользователя, о какой-то ошибке.

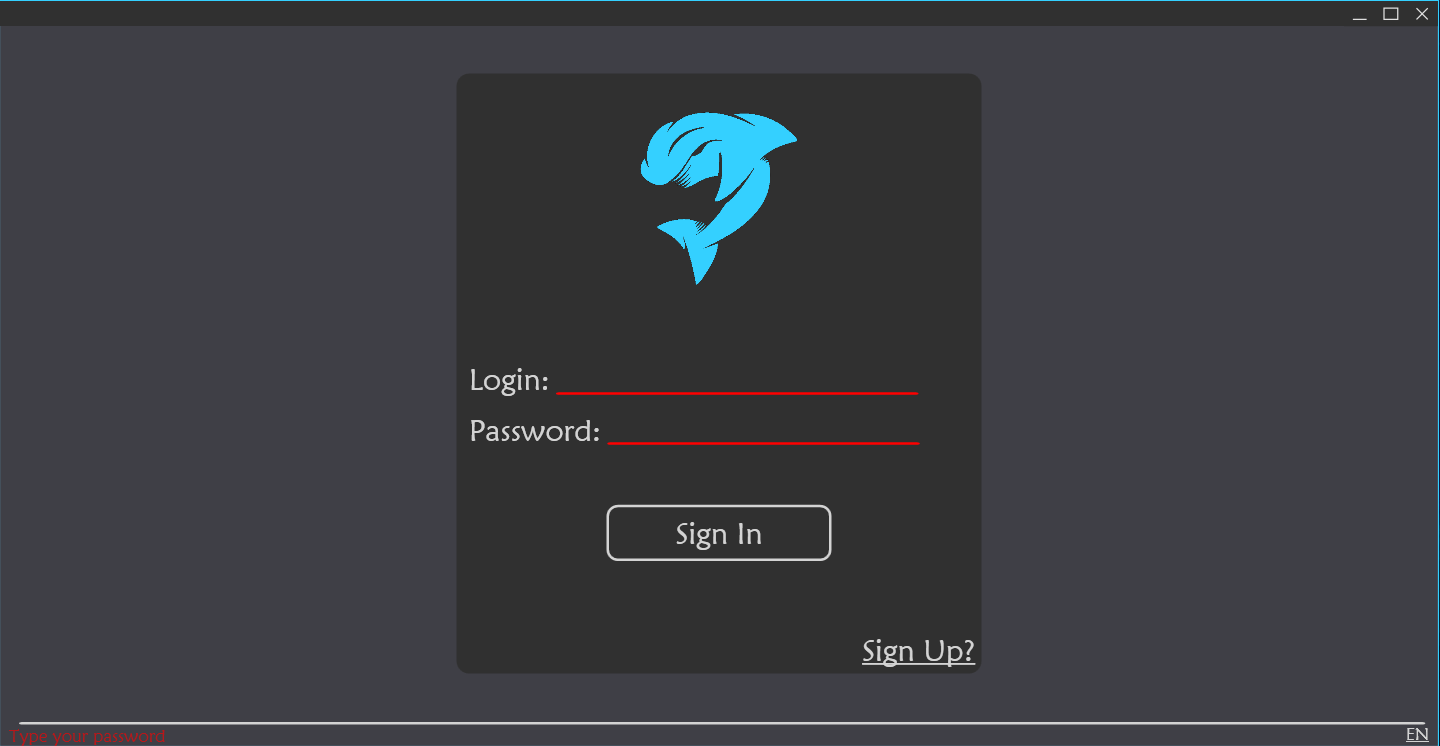


Рисунок 5.1 – Страница авторизации

Такие же действия системы мы можем наблюдать и на странице регистрации, рисунок 5.2.

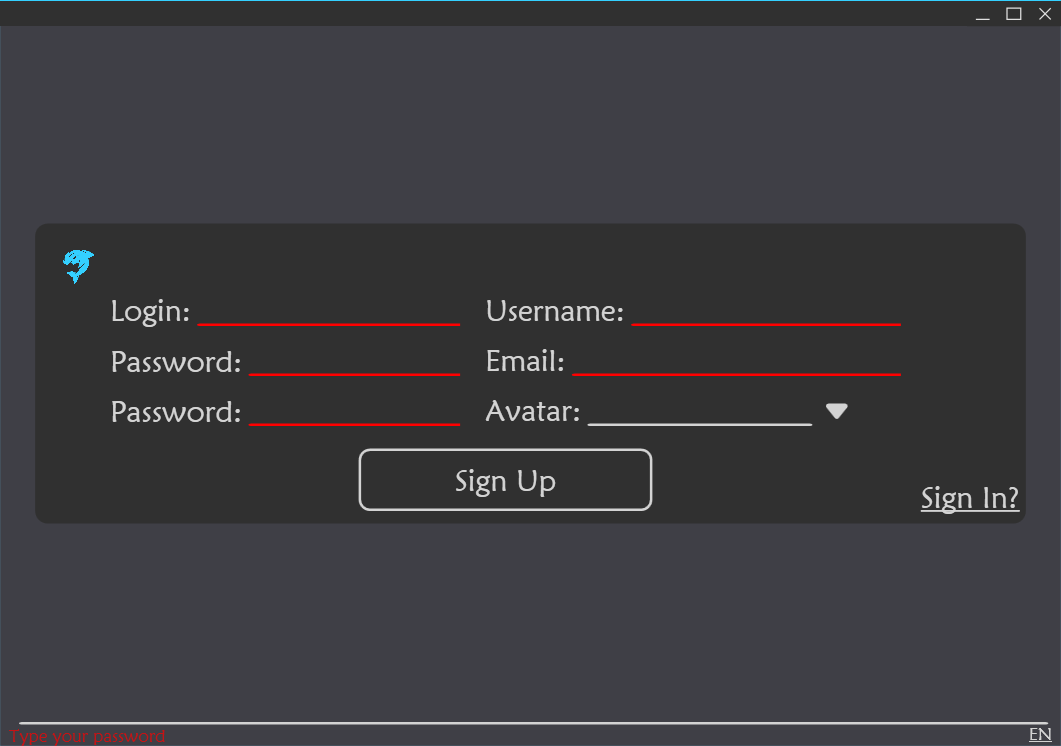


Рисунок 5.2 – Страница регистрации

При отсутствии подлючения к серверу в левом нижнем углу, в строке состояния будет отображаться соответствующее сообщение, рисунок 5.3.

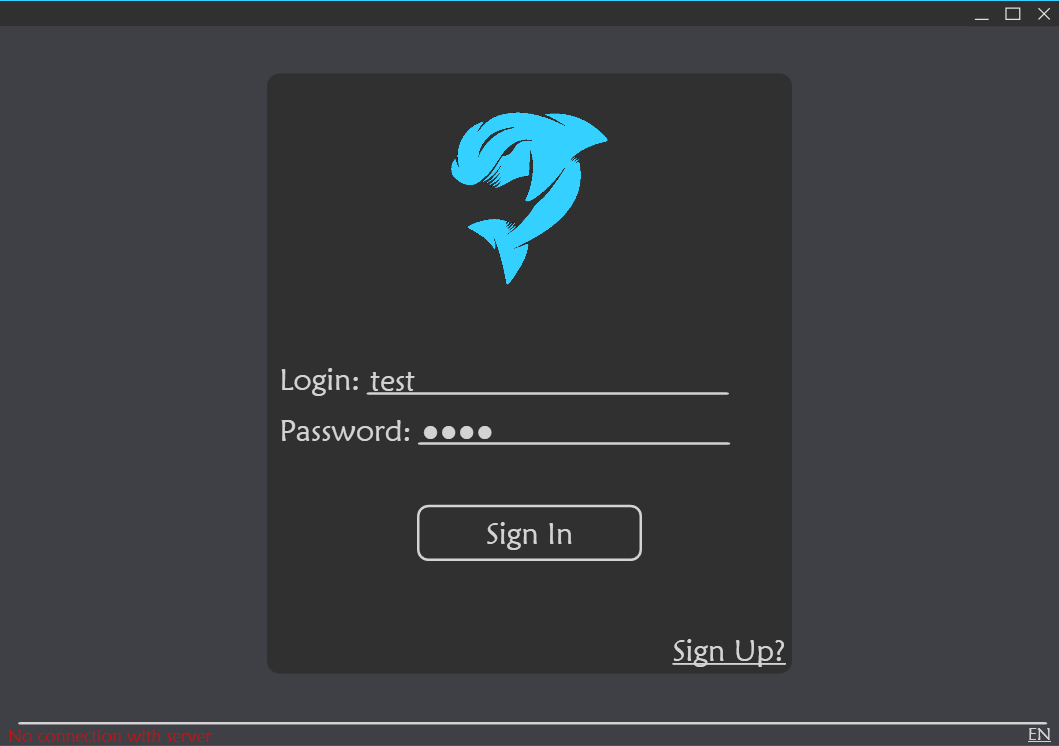


Рисунок 5.3 – Уведомление об отсутствии подключения к серверу

Валидация данных в приложении предусмотренна таким образом, что пользователю просто не предоставляется возможность ввести некорректные данные, так же если он забудет заполнить какое-то поле, то ему придет уведомление в строке состояния.

# **Руководство пользователя**

После запуска приложения пользователя встречает страница авторизации, она представлена на рисунке 6.1.

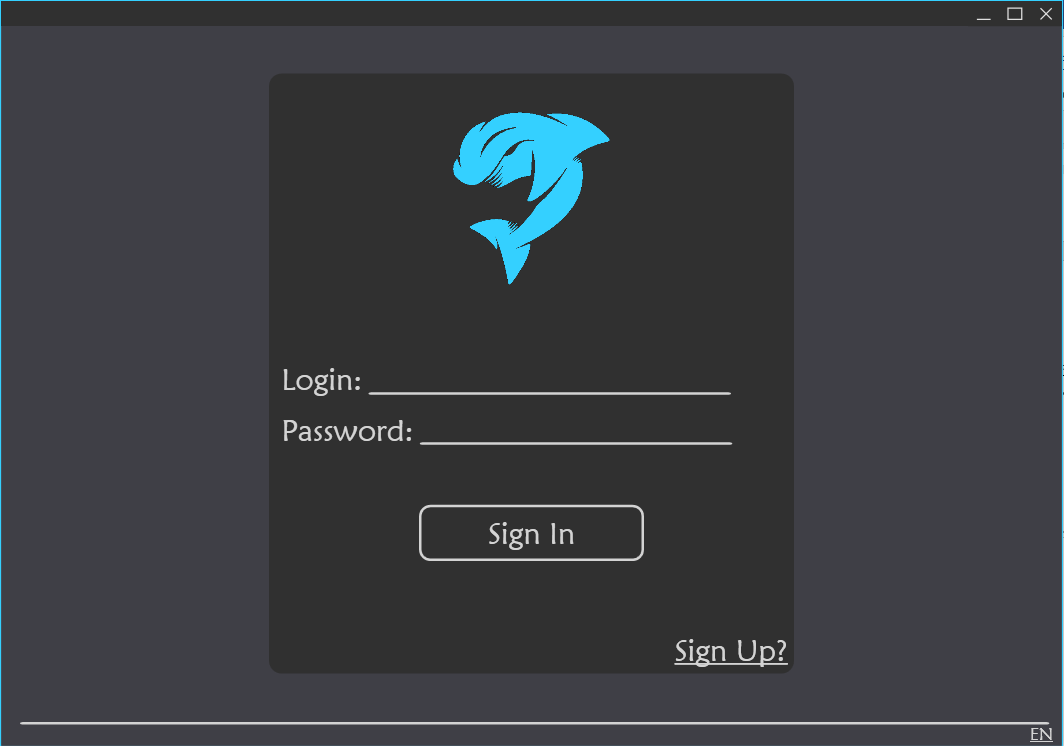


Рисунок 6.1 – Страница авторизации

В случае если пользователь не имеет аккаунт, ему необходимо нажать кнопку «Зарегистрироваться?», после чего он увидит страницу регистрации, где пользователю будет необходимо ввести информацию о себе. Окно регистрации представлено на рисунке 6.2.

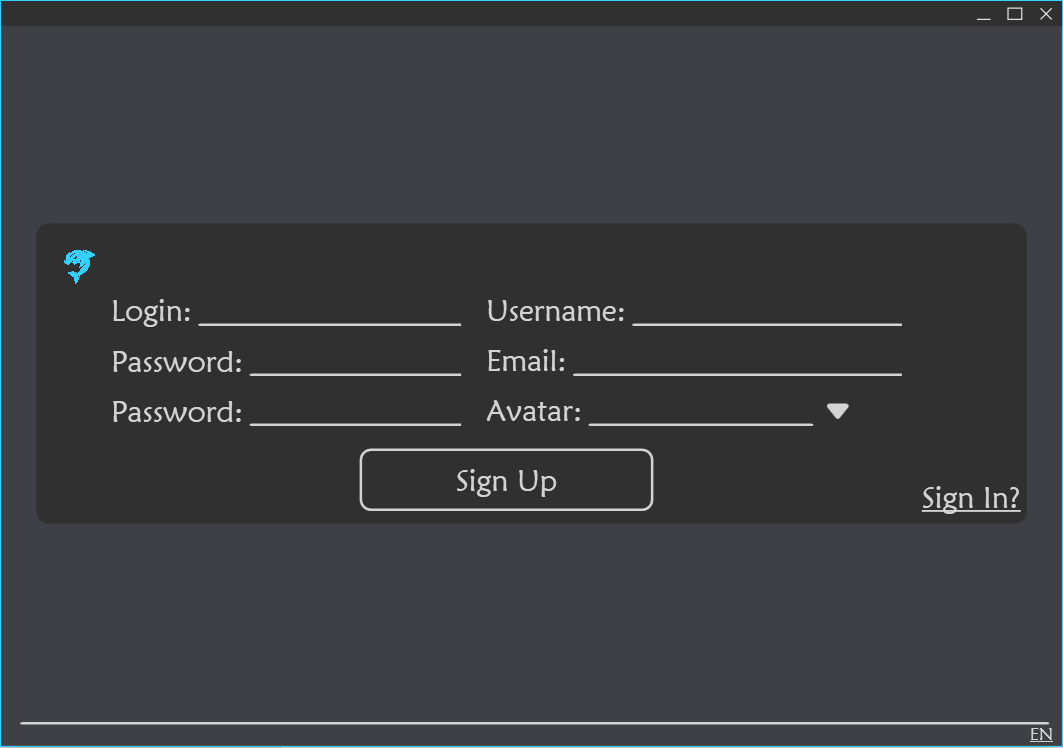


Рисунок 6.2 – Страница авторизации

После успешной авторизации или регистрации пользователю будет доступно основная страница приложения, рисунок 6.3.

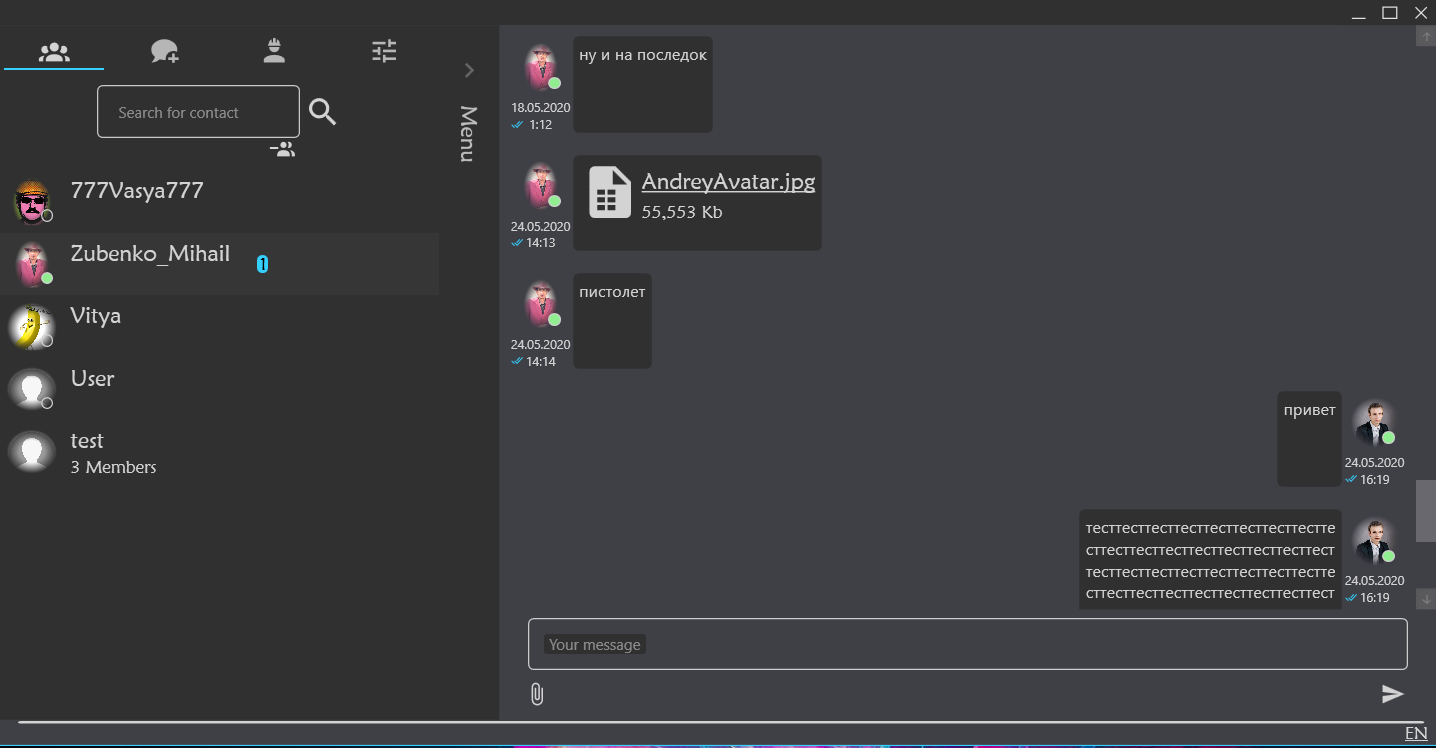


Рисунок 6.3 – Основная страница приложения

Здесь доступно 3 обязательные вкладки и 1 дополнительная, если у пользователя есть права админа, то появится вкладка «Панель Администратора». А панели контактов можно выбрать текущий чат, удалить контакт из своей сети, найти новый контакт вне своей сети, выйти из беседы, каждое такое действие сопровождается подтверждающим диалоговым окном, рисунок 6.4.

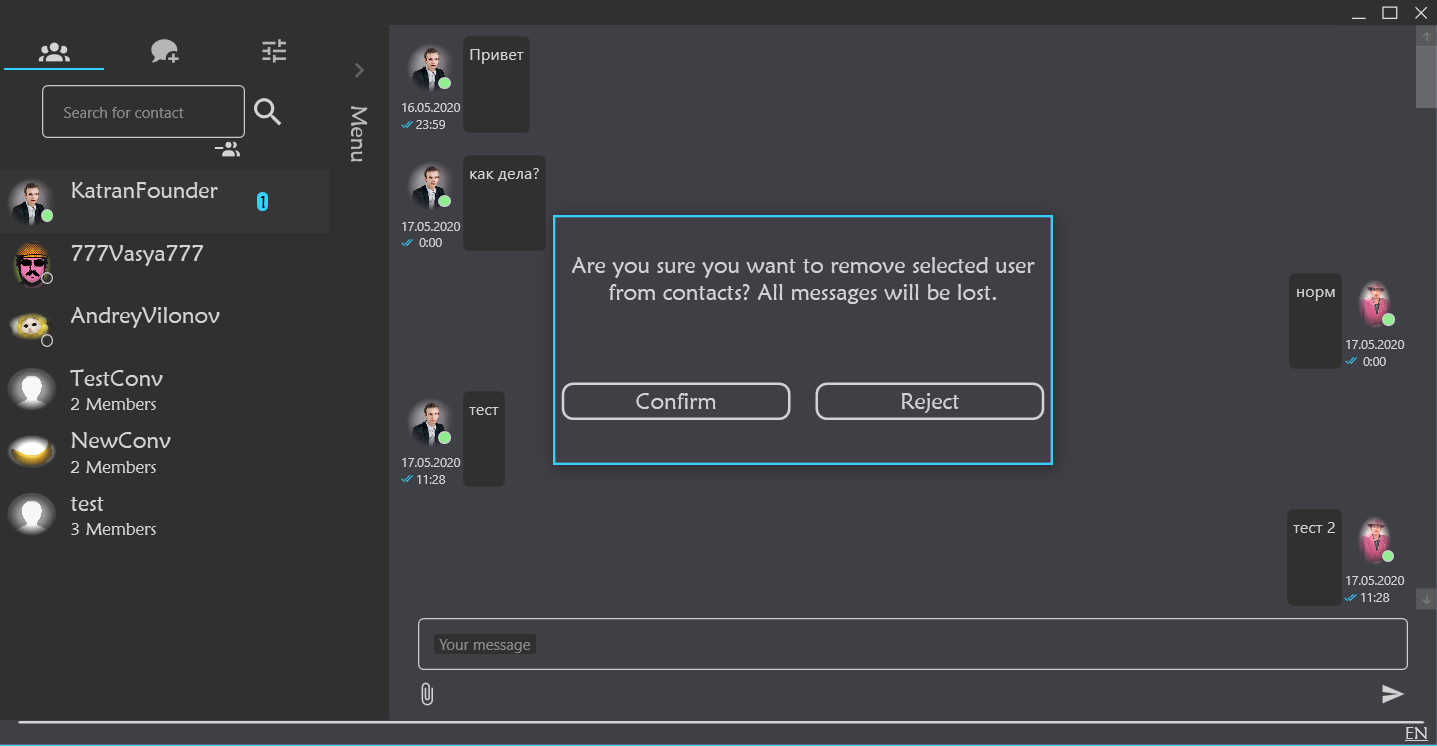


Рисунок 6.4 – Модальные диалоговые окна

Во вкладке «Создание Беседы» пользователь может создать беседу и добавить туда людей из своих контактов, после того, как беседа будет создана во вкладке «Контакты» появится соответствующий элемент, рисунок 6.5.

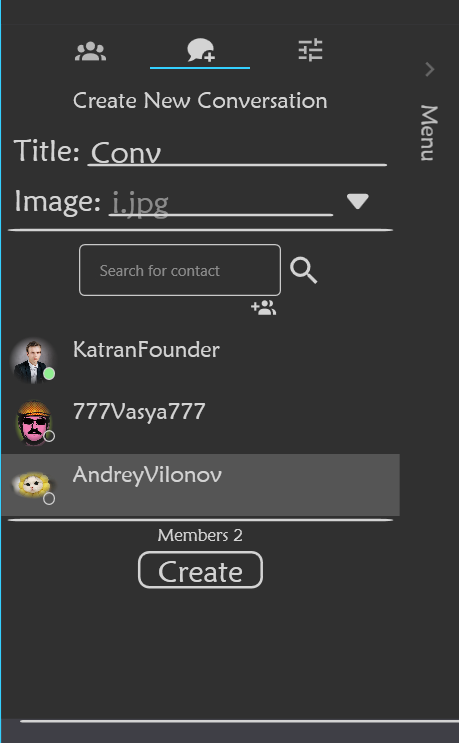


Рисунок 6.5 – Создание бесед

Во вкладке администратора можно найти пользователя по его нику в приложении и разблокировать или заблокировать, это сразу же отобразится у него на экране и в сети контактов у других пользователей он будет помечен как «Заблокирован», рисунок 6.5, 6.6 и 6.7 соответственно.

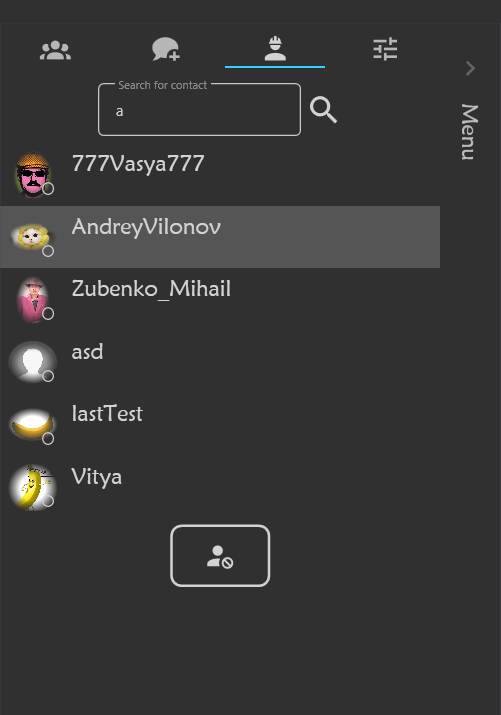


Рисунок 6.5 – Вкладка блокировки пользователей

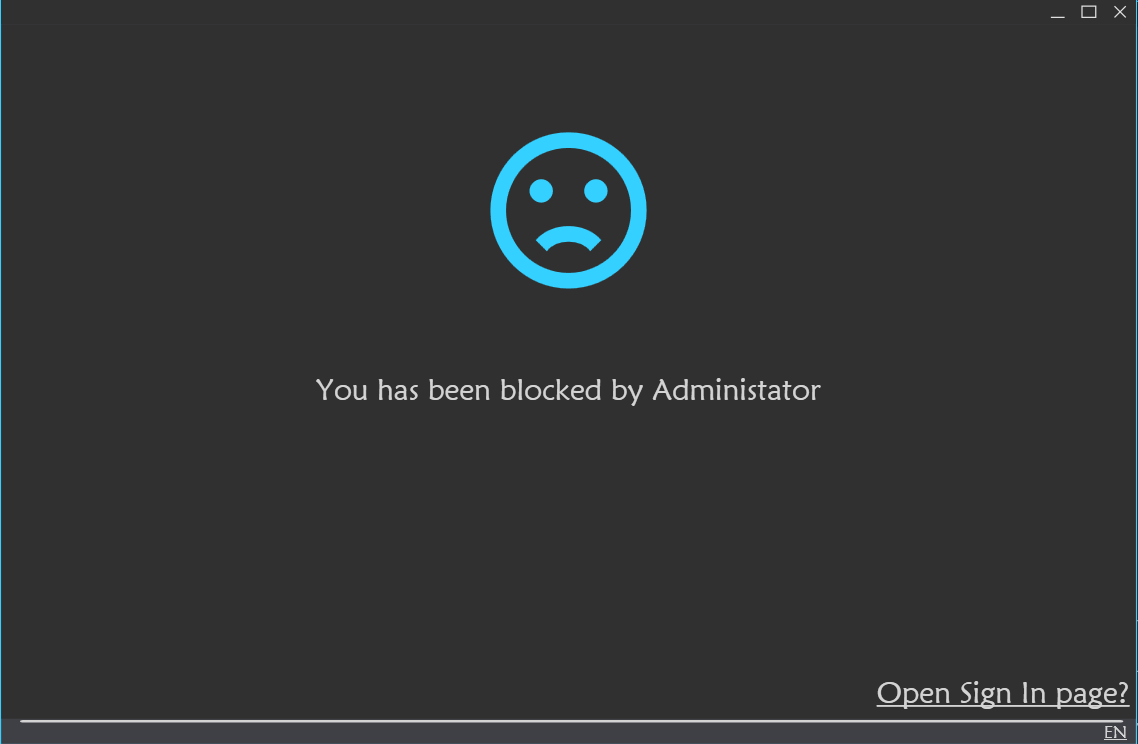


Рисунок 6.6 – Страница, появляющаяся у заблокированных пользователей

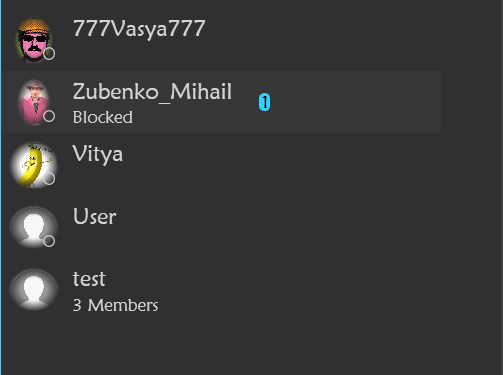


Рисунок 6.7 – Отображение в сети контактов, что пользователь заблокирован

Вкладка «Настройки», здесь можно сменить тему приложения и язык, все изменения сохранятся и при следующем входе будет так, как было настроено. Так же присутствует кнопка выхода из аккаунта, рисунок 6.8.

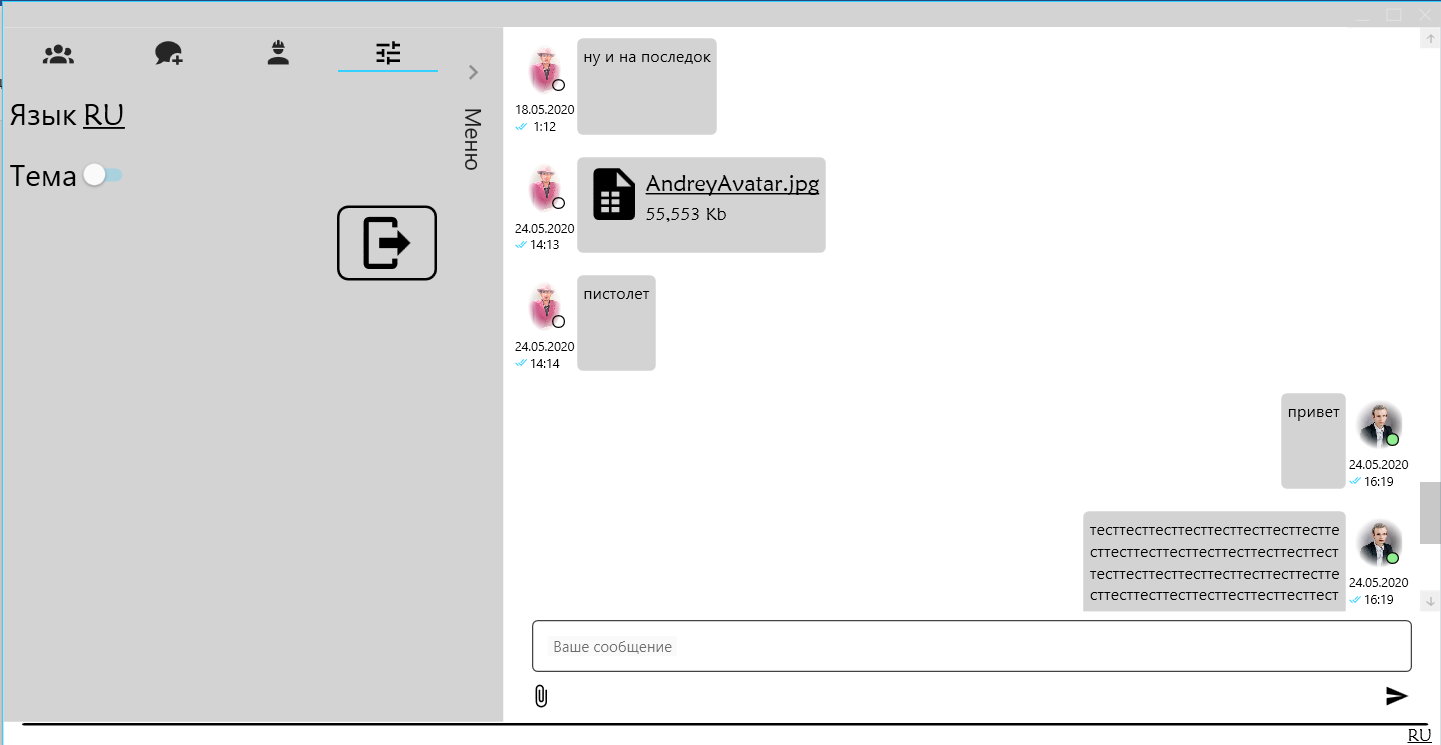


Рисунок 6.8 – Вкладка настроек

В чате отображается не только текст сообщения, но и аватарка отправителя, его статус(онлайн/оффлайн), состояние сообщения(не отправлено/отправлено/прочитано), время. Так же присутствует возможность обмена файлами. При наведении на аватарку собеседника будет указан его ник в приложении, рисунок 6.9. Чтобы сообщение засчиталось как прочитанное на него нужно навестись мышкой. При нажатии на «скрепку» предложится выбрать файл для отправки, после выбора он сразу же отправится в чат. Если нужно скачать файл из чата, то достаточно нажать на имя файла и выбрать куда сохранить.

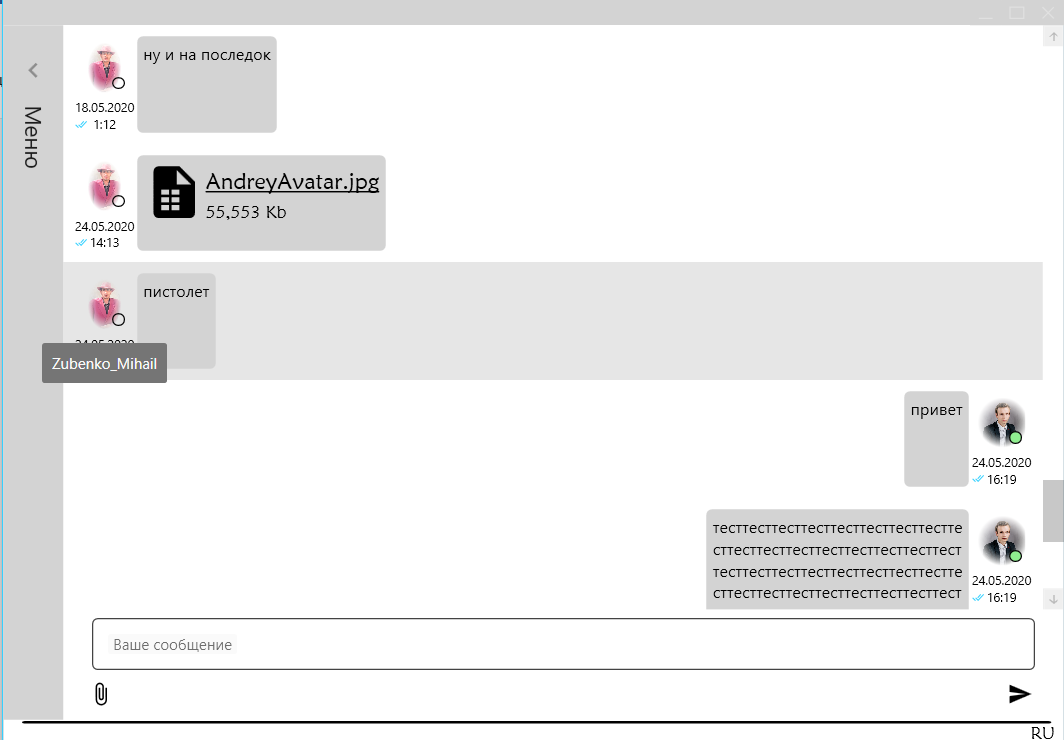


Рисунок 6.9 – Чат и его возможности

# **ЗАКЛЮЧЕНИЕ**

В процессе решения поставленной задачи была достигнута поставленная цель по созданию приложения «Мессенджер». При разработке были выполнены все пункты из указанного списка предполагаемого основного функционала приложения, кроме отправки сообщения всем пользователям. Таким образом, была достигнута цель и создан мессенджер.

В программном средстве были реализованы следующие функции:

* Авторизация и регистрация;
* Отправка сообщений/файлов и просмотр сообщений;
* Отображение статуса сообщений и пользователей;
* Добавление/удаление контактов из сети;
* Блокировка/разблокировка пользователей;
* Смена темы и языка приложения;
* Отображение количества непрочитанных сообщений.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объеме.

# **Список использованных источников**

1. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
2. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с.
3. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: <http://msdn.microsoft.com/library/rus/>. Дата доступа: 20.04.2018
4. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>. Дата доступа: 21.05.2018
5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> Дата доступа: 13.04.2018
6. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд. — СПб.: Питер, 2013. — 896 с.: ил. — (Серия «Мастер-класс»).