МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-98 01 03 «Программное обеспечение информационной безопасности мобильных систем»

Направление специальности 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем (программирование интернет-приложений)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Современные технологии программирования мобильных систем»

Тема: Программное средство «Мессенджер»

Исполнитель

студентка 2 курса 7 группы Курносенко Софья Андреевна

(Ф.И.О.)

Руководитель ассистент Панченко О. Л.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой ­

Руководитель Панченко О. Л.

(подпись)

# Введение

Социальные сети на сегодняшний день стали одним из самых популярных сервисов, удерживающих внимание большей части интернет-аудитории. Актуальность данной темы связана с интенсивным развитием новых информационных технологий и Интернет, растущими тенденциями к компьютеризации различных сфер жизни. Особо стоит отметить быстрое развитие электронной коммуникации – коммуникации между двумя или более людьми, осуществляемой в электронной среде: в Интернете или локальной сети.

В качестве интерфейса прикладного программирования был выбран API-интерфейс — Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом.

Для работы с WPF использовался объектно-ориентированный язык программирования С#.

Хранение данных осуществляется в Microsoft SQL Server.

Для успешной реализации курсового проекта необходимо:

* провести анализ соответствующей литературы;
* ознакомиться с прототипами программных средств выбранной мной темы;
* определить функциональные требования;
* продумать структуру базы данных;
* продумать структуру проекта;
* реализовать программное средство;
* протестировать программное средство;
* написать руководство пользователя.

Содержание данной пояснительной записки отражает все этапы выполнения моего курсового проекта.

## Аналитический обзор литературы и формирование требований

## Анализ прототипов

Были проанализированы цели и задачи, поставленные в данном курсовом проекте, а также рассмотрены аналогичные примеры их решений. На основании анализа всех достоинств и недостатков данных альтернативных решений были сформулированы требования к данному программному средству.

1.1. Instagram

Instagram — американская социальная сеть для обмена фотографиями и видео, основанная в 2010 году Кевином Систромом и Майком Кригером, а затем приобретенная Facebook Inc. Приложение позволяет пользователям загружать медиафайлы, которые можно редактировать с помощью фильтров и упорядочивать по хэштегам и географическим меткам. Публикациями можно делиться публично или с предварительно одобренными подписчиками. Пользователи могут просматривать контент других пользователей по тегу и местоположению, просматривать трендовый контент, например фотографии, и подписываться на других пользователей, чтобы добавить свой контент в личную ленту.

Интерфейс приложения представлен на рисунке 1.1.

Достоинства приложения:

- создание снимков и видеофайлов;

- обработка фотографий с помощью специализированных уникальных фильтров;

- загрузка изображений в Интернет и публикации в своем профиле;

- моментальный перепост сообщений в другие социальные сети;

- просмотр фотографий других пользователей;

- поиск аккаунтов;

- общение в комментариях к фото.

Недостатки приложения:

- низкие возможности в плане настройки приватности;

- отсутствие полноценного сайта для загрузки фото;

- техподдержка;

- невозможность скачивания фотографий.

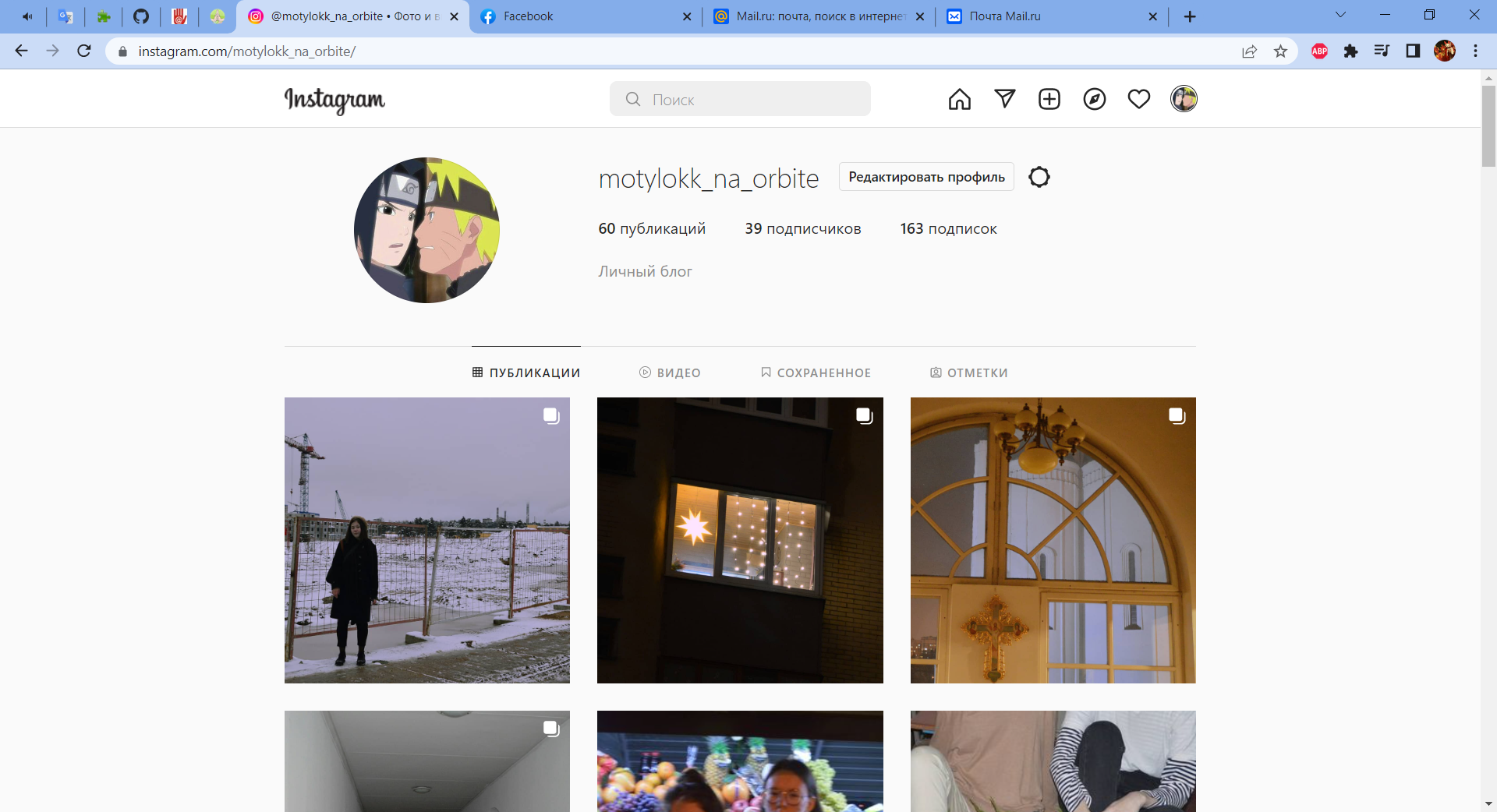


Рисунок 1.1 — Скриншот приложения Instagram

1.1.2 Facebook

Facebook — крупнейшая социальная сеть в мире, которой владеет компания Meta. Facebook позволяет создать профиль с фотографией и без неё и информацией о себе, приглашать друзей, обмениваться с ними сообщениями, изменять свой статус, оставлять сообщения на своей и чужой «стенах», загружать фотографии и видеозаписи, создавать группы (сообщества по интересам).

Интерфейс приложения представлен на рисунке 1.2.

Достоинства приложения:

– фильтрация контента в новостной ленте

– большой набор функций рекламного кабинета

– привязка профилей Facebook и Instagram

– индексация страниц поисковыми системами

Недостатки приложения:

– недостаточная защита персональных данных пользователей

– запрет на использование определенных формулировок;

– недопустима дискриминация по любому признаку;

– строгий регламент заголовков рекламных постов и оформления баннеров

– непонятный алгоритм отбора публикация при формировании новостной ленты

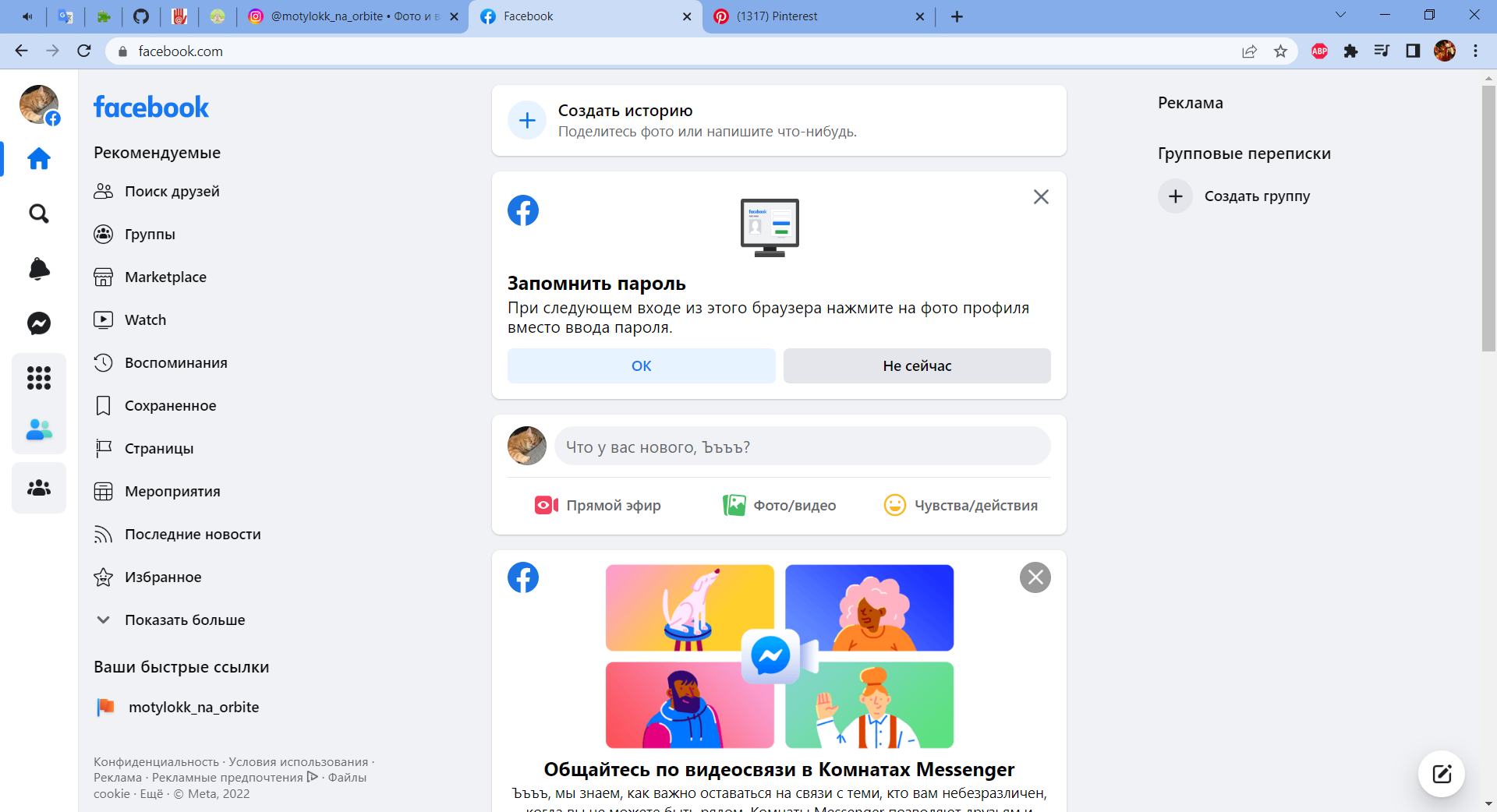


Рисунок 1.2 — Скриншот приложения Facebook

* 1. Требования к проекту

Проанализировав вышеперечисленные прототипы социальных сетей, я выделила основную задачу и функциональные требования проекта.

Моей задачей является разработка десктопного приложения, позволяющее обмениваться сообщениями между пользователями, регистрироваться/авторизоваться в приложении, редактировать пользователем свой профиль и т.д. Программное средство должно содержать профиль авторизованного пользователя с его контактными данными. Интерфейс должен быть простым и удобным для использования.

Функциональные требования описаны в главе 2.

# 2. Анализ требований к программному средству и разработка функциональных требований

## 2.1. Описание средств разработки

При разработке приложения были использованы:

-интегрированная среда разработки Microsoft VS 2022;

- программная платформа .NET Framework 6.0;

- язык программирования C#;

- технологии WPF;

- расширяемый язык разметки XAML;

- технология Entity Framework;

- MS SQL Server;

- платформа для разработки сервисов WCF.

### 2.1.1. Microsoft Visual Studio 2022

Microsoft Visual Studio 2022 — это интегрированная среда разработки для написания, отладки и сборки кода, а также последующей публикации приложений. Данный продукт позволяет разрабатывать не только консольные, но и десктопные приложения, с использованиям таких технологий, как WinForms или WPF.

### 2.1.2. Программная платформа .NET Framework 6.0

Платформа .NET Framework — это созданная в 2002 году технология Microsoft, которая поддерживает создание и выполнение веб-служб и приложений Windows, обеспечивающая согласованную объектно-ориентированной среду программирования для локального сохранения и выполнения объектного кода, локального выполнения кода, распределенного в Интернете, либо для его удаленного выполнения. Основой платформы является общеязыковая среда Common Language Runtime, выполняющая код на разных поддерживаемых языках программирования.

### 2.1.3. Язык программирования C#

В качестве языка программирования используется C# – основной язык разработки в .NET Framework. Язык объектно-ориентированный, имеет строгую статическую типизацию, поддерживает перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения. Используется как основной язык в технологии WPF.

### 2.1.4. Технология WPF

Для предоставления пользовательского интерфейса и разграничения дизайна и бизнес-логики используется технология Microsoft WPF – аналог WinForms, система для построения клиентских приложений Windows с возможностями взаимодействия с пользователем и графическая подсистема в составе .NET Framework, использующая язык разметки XAML.

### 2.1.5. Расширяемый язык разметки XAML

WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language) элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. XAML представляет собой язык декларативного описания интерфейса, основанный на XML.

### 2.1.6. Технология Entity Framework

Для осуществления связи между базой данных и приложением на C# необходим посредник, которым будет являться технология Entity Framework. Она основана на платформе .NET Framework и предназначена для работы с данными.

Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища.

* Entity Framework предполагает три возможных способа взаимодействия с базой данных:Database First: Entity Framework Core создает набор классов, которые отражают модель конкретной базы данных;
* Model First: сначала разработчик создает модель базы данных, по которой затем Entity Framework Core создает реальную базу данных на сервере;
* Code First: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework Core по этой модели генерирует базу данных и ее таблицы.

### 2.1.7. Технология WCF

Windows Communication Foundation — программный фреймворк, используемый для обмена данными между приложениями, входящий в состав .NET Framework. WCF делает возможным построение безопасных и надёжных транзакционных систем через упрощённую унифицированную программную модель межплатформенного взаимодействия.

2.1.8. MS SQL Server

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта. Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос 10 на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения. Основной используемый язык запросов — Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

2.2. Спецификация функциональных требований к программному средству

Программное средство должно предоставлять следующие функциональные возможности:

Для пользователя:

- Выполнять регистрацию и авторизацию;

- Вести чат с отдельным пользователем в режиме реального времени;

- Поиск пользователя по имени для начала беседы с ним;

- Удаление своего профиля.

2.3. Спецификация функциональных требований

Для функциональности программного средства необходимо создание базы данных для хранения информации приложения. Подробно база данных описано в главе 3.

В программном средстве необходимо реализовать регистрацию и авторизацию пользователей для доступа ко всем возможностям приложения. Для авторизации входными параметрами являются логин и пароль пользователя, которые содержатся в базе данных. Для регистрации входными данными являются логин, пароль, e-mail. Введенные данные, успешно прошедшие валидацию, заносятся в базу данных.

Пользователь может писать сообщения в общий чат, получать сообщения от других пользователей в режиме реального времени. Пользователь имеет возможность просматривать информацию о других пользователях. Также есть возможность редактировать профиль, в том числе добавлять фотографию на аватар. В случае необходимости пользователь может удалить свой аккаунт, но ранее написанные им сообщения удалены не будут.

# Проектирование программного средства

* 1. Общая структура

Программное средство «Мессенджер» имеет следующую структуру, представленную на рисунке 3.1.

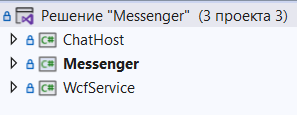


Рисунок 3.1 – Структура решения

Описание структуры проектов решения представлено в таблице 3.1.

Таблица 3.1 – Описание структуры проектов решения Messenger

|  |  |
| --- | --- |
| Имя проекта | Содержание |
| ChatHost | Хост, на котором будет размещаться сервис. |
| Messenger | Клиент, то есть часть приложения, непосредственно с которым будет работать пользователь. |
| WcfService | Сервис (сервер), предоставляющий клиенту услуги обмена сообщениями. |

Проект ChatHost имеет структуру представленную на рисунке 3.2.

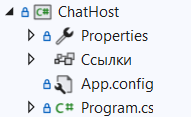


Рисунок 3.2 – Структура проекта ChatHost

Описание структуры основных файлов проекта ChatHost представлено в таблице 3.2.

Таблица 3.2 – Описание структуры проекта ChatHost

|  |  |
| --- | --- |
| Имя файла/папки | Содержание |
| App.config | Файл с параметрами проекта. |
| Program.cs | Класс, где происходит запуск работы хоста. |

Проект Messenger имеет структуру представленную на рисунке 3.3.

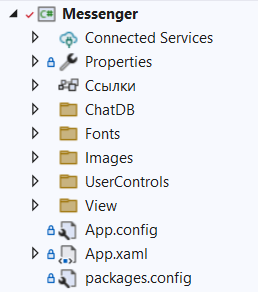


Рисунок 3.3 – Структура проекта Messenger

Описание структуры основных файлов и папок проекта Messengerпредставлено в таблице 3.3.

Таблица 3.3 – Описание структуры проекта Messenger

|  |  |
| --- | --- |
| Имя файла/папки | Содержание |
| Connected Services | Подключенные службы — это набор инструментов в Visual Studio, которые позволяют подключить приложение к конечным точкам wcf.  Имеет ссылку на сервис, к которому будет подключаться. |
| Папка ChatDB | Папка, содержащая классы, необходимые для взаимодействия с Entity Framework, то есть сущности. |
| Папка Fonts | Шрифты, используемые в приложении. |
| Папка Images | Изображения, используемые в приложении. |
| Папка UserControls | Пользовательские элементы управления. |
| Папка View | Все окна и страницы, а также стили и шаблоны приложения. |
| App.xaml | Стартовая точка приложения. |

Более подробная структура содержимого папок проекта Messenger представлена на рисунке 3.4.

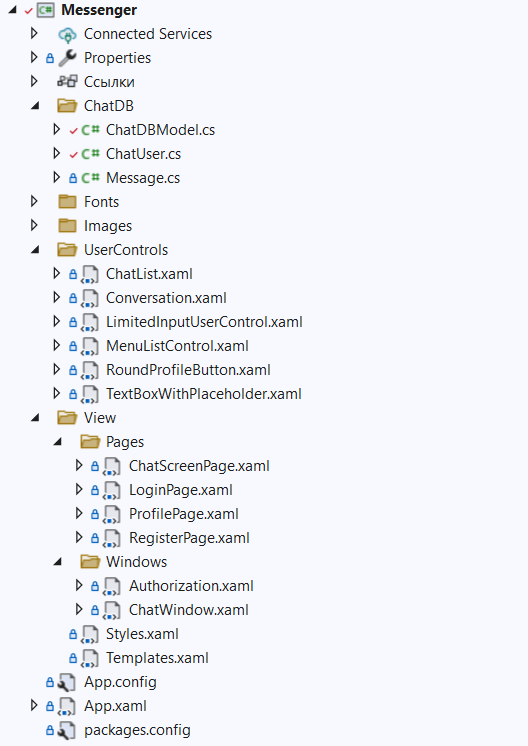


Рисунок 3.4 – Подробная структура проекта Messenger

Описание структуры основных файлов и папок проекта Messengerпредставлено в таблице 3.3.

Таблица 3.3 – Описание структуры проекта Messenger

|  |  |
| --- | --- |
| Имя файла/папки | Содержание |
| ChatDbModel.cs | Класс контекста подключения к базе данных. |
| ChatUser.cs | Класс сущности ChatUser. |
| Message.cs | Класс сущности Message. |
| ChatList.xaml | Пользовательский элемент управления, определяющий стиль ListBox’а с зарегистрировавшимися пользователями. |
| Conversation.xaml | Пользовательский элемент управления, определяющий стиль ListBox’а с сообщениями. |
| LimitedInputUserControl.xaml | Пользовательский элемент управления, который представляет из себя поле ввода с ограничением на ввод определенного количества символов и отображением количества уже введенных символов. |
| MenuListControl.xaml | Пользовательский элемент управления, определяющий стиль ListBox’а с иконками меню. |
| RoundProfileButton.xaml | Пользовательский элемент управления, определяющий стиль отображения картинки пользователя. |
| TextBoxWithPlaceHolder.xaml | Пользовательский элемент управления, определяющий стиль TextBox’а для ввода данных в формах авторизации. |
| ChatScreenPage.xaml | Страница с чатом. |
| LoginPage.xaml | Страница входа. |
| ProfilePage.xaml | Страница профиля. |
| RegisterPage.xaml | Страница регистрации. |
| Authorization.xaml | Окно авторизации (будет отображать в зависимости от выбора пользователя LoginPage.xaml или RegisterPage.xaml). |
| ChatWindow.xaml | Окно чата (будет отображать в зависимости от выбора пользователя ProfilePage.xaml или ChatScreenPage.xaml). |
| Styles.xaml | Стили, используемые в проекте. |
| Templates.xaml | Шаблоны, используемые в проекте. |

Проект WcfService имеет структуру представленную на рисунке 3.5.

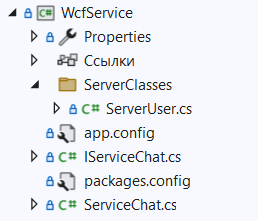


Рисунок 3.5 – Структура проекта WcfService

Описание структуры основных файлов проекта **WcfService** представлено в таблице 3.5.

Таблица 3.5 – Описание структуры проекта WcfService

|  |  |
| --- | --- |
| Имя файла | Содержание |
| app.config | Файл конфигурации сервиса. |
| ServerUser.cs | Класс пользователей сервиса. |
| IServiceChat.cs | Контракт службы. В контракте службы указывается ее сигнатура, отправляемые и получаемые ей данные и прочие данные, требуемые контрактом.  То есть это интерфейс с услугами, предоставляемыми сервисом клиенту. |
| ServiceChat.cs | Реализация контракта. |

* 1. Взаимоотношения между классами

Для визуализации взаимосвязей между классами используется диаграмма UML – графическое представление набора элементов, изображаемое чаще всего в виде связанного графа с вершинами (сущностями) и ребрами (отношениями).

Для представления внутренней структуры программы в виде классов и связей между ними используется диаграмма классов. Приложение спроектировано таким образом, что каждый класс выполняет свои функции и практически не зависит от других. Диаграмма классов представлена в Приложении А.

* 1. Модель базы данных

Для реализации поставленной задачи была создана база данных Chat. Для ее создания использовалась система управления реляционными базами данных MSQL Server. База данных состоит из таблиц, представленных на рисунке 3.6. Скрипт для создания базы данных представлен в приложении Б.

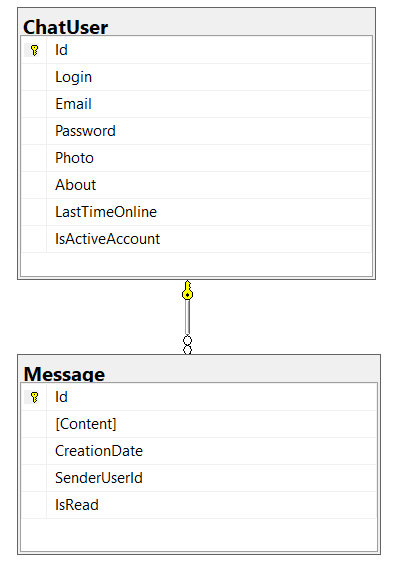


Рисунок 3.6 – База данных Chat

На рисунке 3.7 проиллюстрирована структура таблицы «ChatUser», которая содержит информацию о пользователях. Поля таблицы:

- Id – уникальный идентификатор пользователя.

- Login – логин пользователя.

- Email – адрес электронной почты пользователя.

- Password – пароль.

- Photo – путь к фотографии пользователя.

- About – описание пользователя («о себе»).

- LastTimeOnline – дата последнего пребывания пользователя в сети.

- IsActiveAccount – статус аккаунта (1 – активный, 2 – неактивный (удален)).

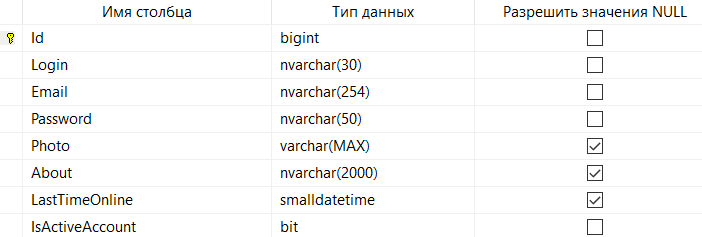


Рисунок 3.7 – Структура таблицы «ChatUser»

На рисунке 3.8 изображена структура таблицы «Message», содержащая информацию о сообщениях, отправленных пользователями. Поля таблицы:

- Id – уникальный идентификатор сообщения.

- Content – содержание текстового сообщения.

- CreationDate – дата создания (отправки) сообщения.

- SenderUserId – идентификатор пользователя, отправившего сообщение.

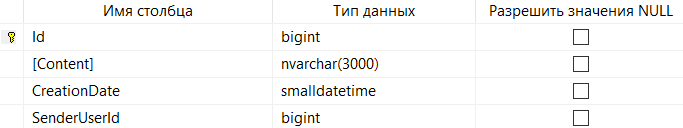


Рисунок 3.8 – Структура таблицы «Message»

* 1. Проектирование архитектуры проекта

Для общего представления функционального назначения системы используется *диаграмма использования*, описывающая, какой функционал разрабатываемой программной системы доступен каждой группе пользователей. На диаграмме использования применяются два типа основных сущностей: варианты использования и группы пользователей.

Разные группы пользователей в диаграмме называются актёрами, и обозначают любые сущности, использующие систему. Любая функция системы называется вариантом использования. Каждый вариант использования обозначает набор действий, который может быть использован актёром для взаимодействии с системой, и определяет набор действий, выполняемых этой системой.

Диаграмма использования представлена в приложении В.

# Реализация программного средства

Следующим этапом разработки приложения является непосредственная реализация программного решения в соответствии с уже сформированными требованиями и шаблонами.

* 1. Реализация сущностей

В соответствии с требованиями в качестве хранилища данных программного средства должна быть база данных, поэтому первым шагом в реализации программы является выбор технологии, позволяющей это осуществить. Выбор остановился на Entity Framework.

При использовании Entity Framework в приложении существует три подхода для организации взаимодействия Entity Framework с базой данных: Code-First, Model-First и Database-First. В своем проекте я сначала создала базу данных, а на её основе программа Visual Studio 2022 создавала сущности.

В приложении Б представлен скрипт создания базы данных Chat.

* 1. Реализация авторизации и регистрации

Для того чтобы пользоваться приложением, необходимо для начала зарегистрироваться в системе.

Если пользователь уже зарегистрирован, то следует войти в систему.

В базе данных хранятся логины и пароли пользователей. Для входа в систему используется проверка на введенные логин и пароль пользователя с

имеющимися в базе данных.

* 1. Отправка и получение сообщений

Для отправки сообщений клиентское приложение использует услуги сервиса, для получения – на стороне клиента реализован коллбэк, то есть действия клиента при обращении к нему сервиса.

* 1. Редактирование профиля

После регистрации пользователь в праве вносить изменения в ранее введенную информацию. Он имеет возможность изменить логин, электронную почту, описание, пароль, а также загрузить аватар, вместо картинки по умолчанию. Помимо этого пользователь может удалить свой аккаунт, при этом все сообщения будут сохранены.

1. **Тестирование, проверка работоспособности и анализ полученных результатов**
   1. Тестирование авторизации и регистрации

В момент авторизации и регистрации возможна ситуация, когда пользователь вводит некорректные данные, например, неверный пароль, незарегистрированный логин, некорректный адрес электронной почты. Такие исключения обрабатываются программным средством с помощью подсветки красными полей с некорректными данными. Примеры обработки разных видов исключительных ситуаций представлены на рисунках 5.1 – 5.3.

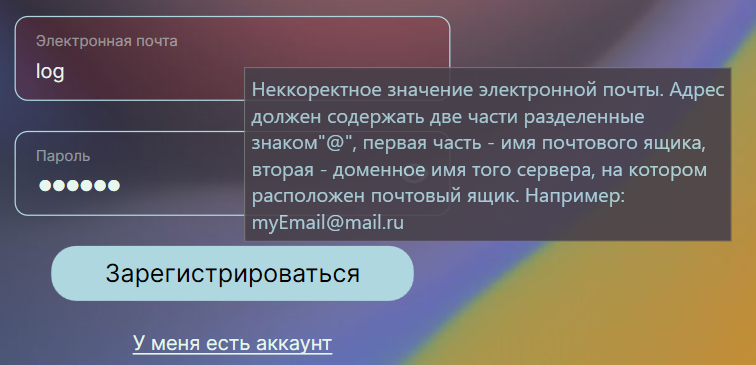


Рис 5.1 – Обработка неправильного e-mail

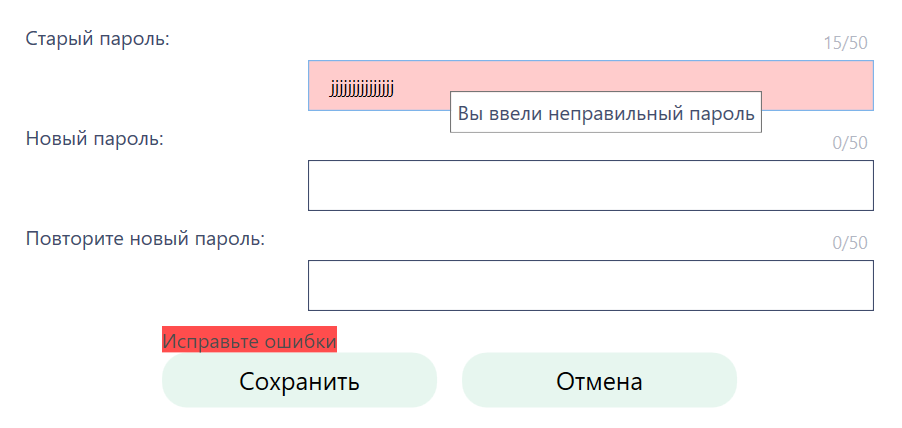


Рис 5.2 – Обработка при вводе неправильного пароля при попытке сменить его

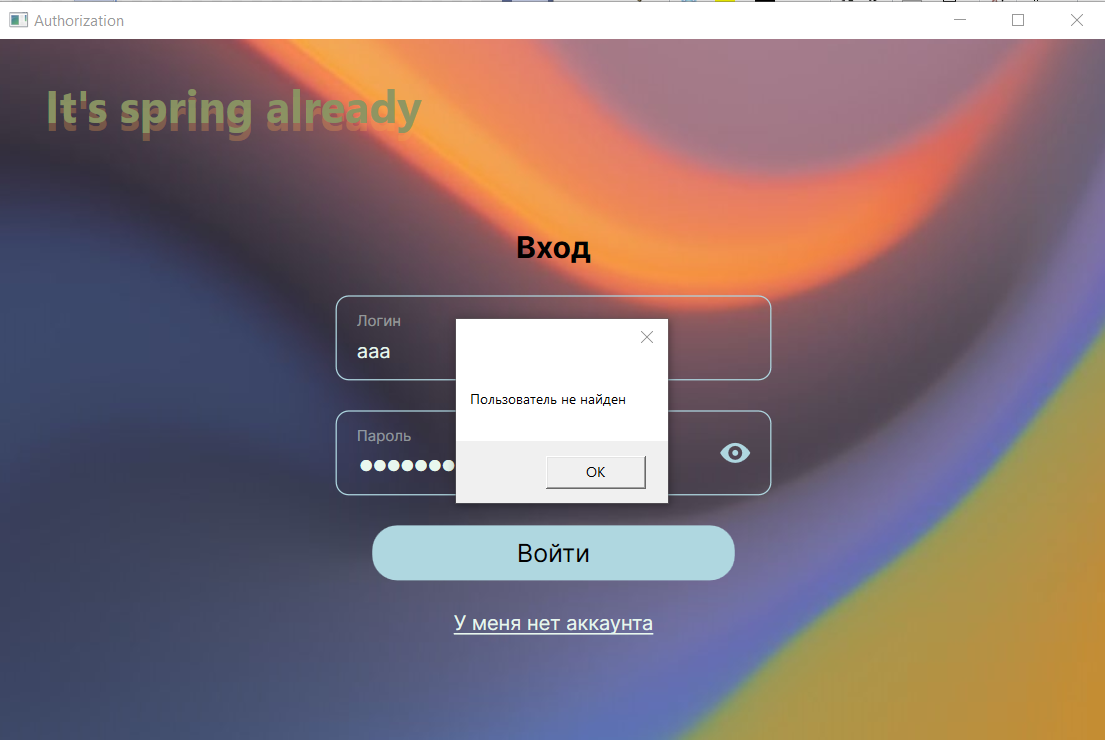


Рис 5.3 – Обработка ошибки при попытке входа незарегистрированного пользователя

5.2. Тестирование отправки сообщений

Пользователь user3 отправил сообщение (рис. 5.4).

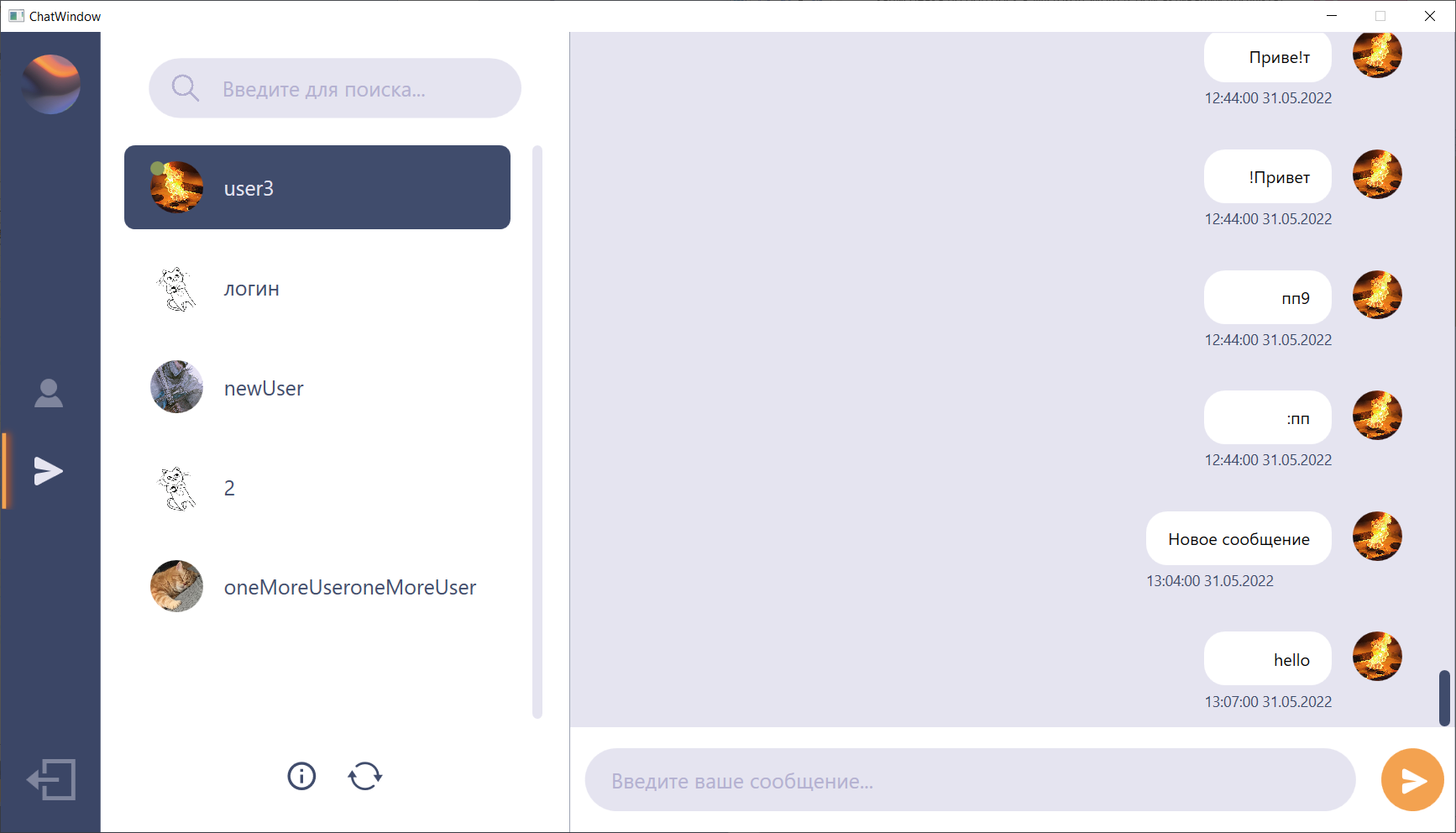


Рис 5.4 – Пользователь отправил сообщение «hello»

Сообщение записано в базу данных (рис. 5.5).

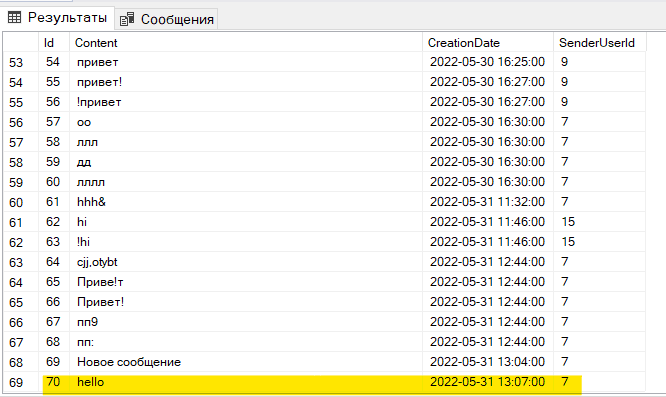


Рис 5.5 – Сохранение отправленных сообщений в базу данных

Остальные пользователи получили сообщение (рис. 5.6).

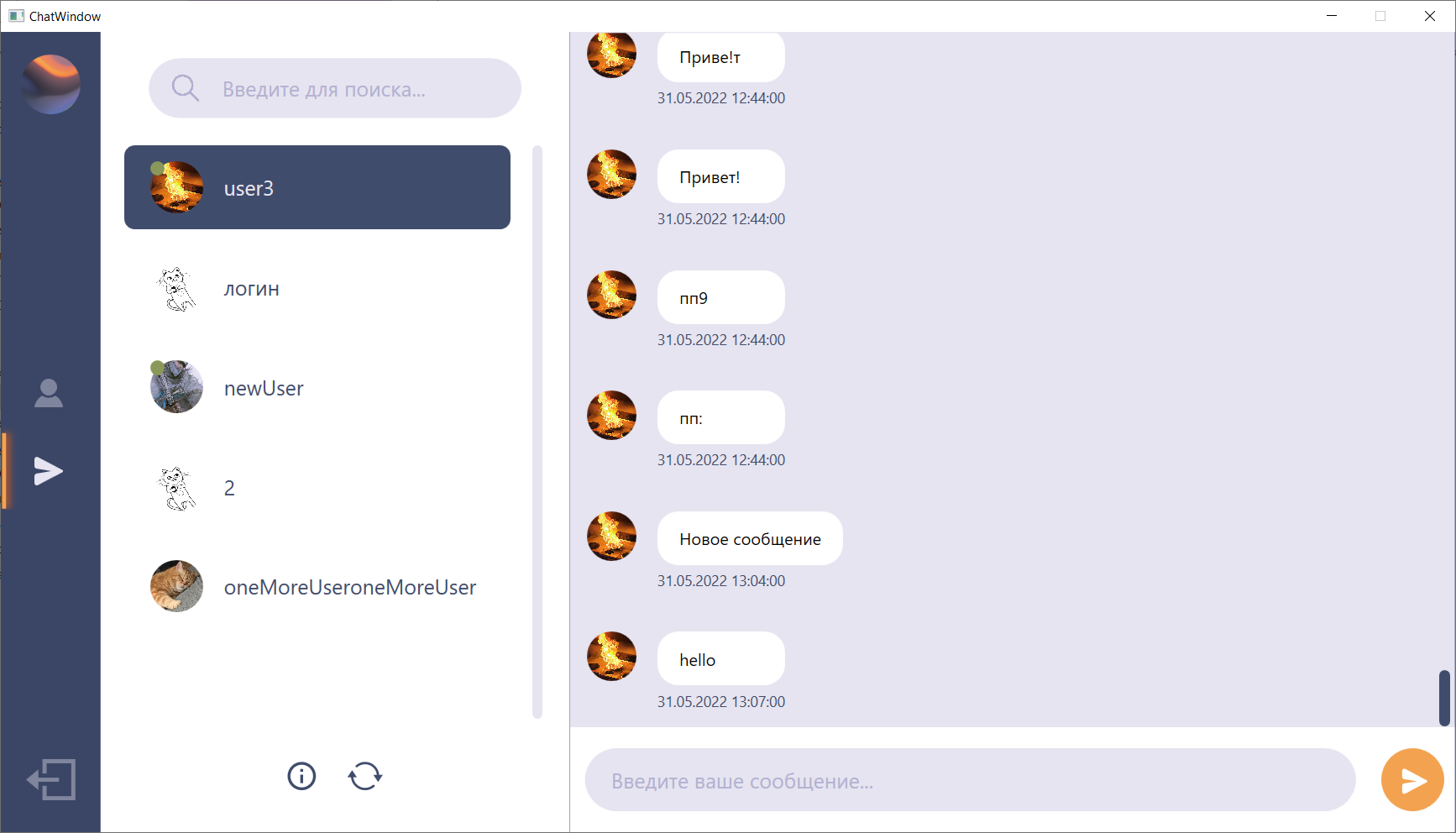


Рис 5.6 – Отображение в общем чате нового сообщения

Заключение

# В ходе выполнения курсовой работы было разработано программное средство «Социальная сеть» на языке C# с использованием технологий Entity Framework, WPF, WCF.

# В ходе выполнения курсовой работы было выполнено следующее:

# разработка базы данных для хранения информации;

# разработка архитектуры приложения;

# разработка функциональной части приложения;

# разработка пользовательского интерфейса;

# написание исходного кода приложения;

# тестирование приложения.

Данное программное средство имеет удобный и понятный интерфейс. Результаты проведенного тестирования показали, что приложение обрабатывает все исключительные ситуации.

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

# Список литературных 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. Мэтью Макдональд — WPF: Windows Presentation Foundation в .NET

4.0 с примерами на C#

1. Язык C# и .NET Framework [Электронный ресурс] — Режим доступа: <http://professorweb.ru/my/csharp/charp_theory/level1/infonet.php>

3. Руководство по ADO.NET Entity Framework 6.0 [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/entityframework/>

ПРИЛОЖЕНИЕ А

Диаграмма классов проекта ChatHost.

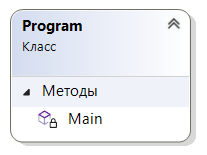


Диаграмма классов проекта Messenger.

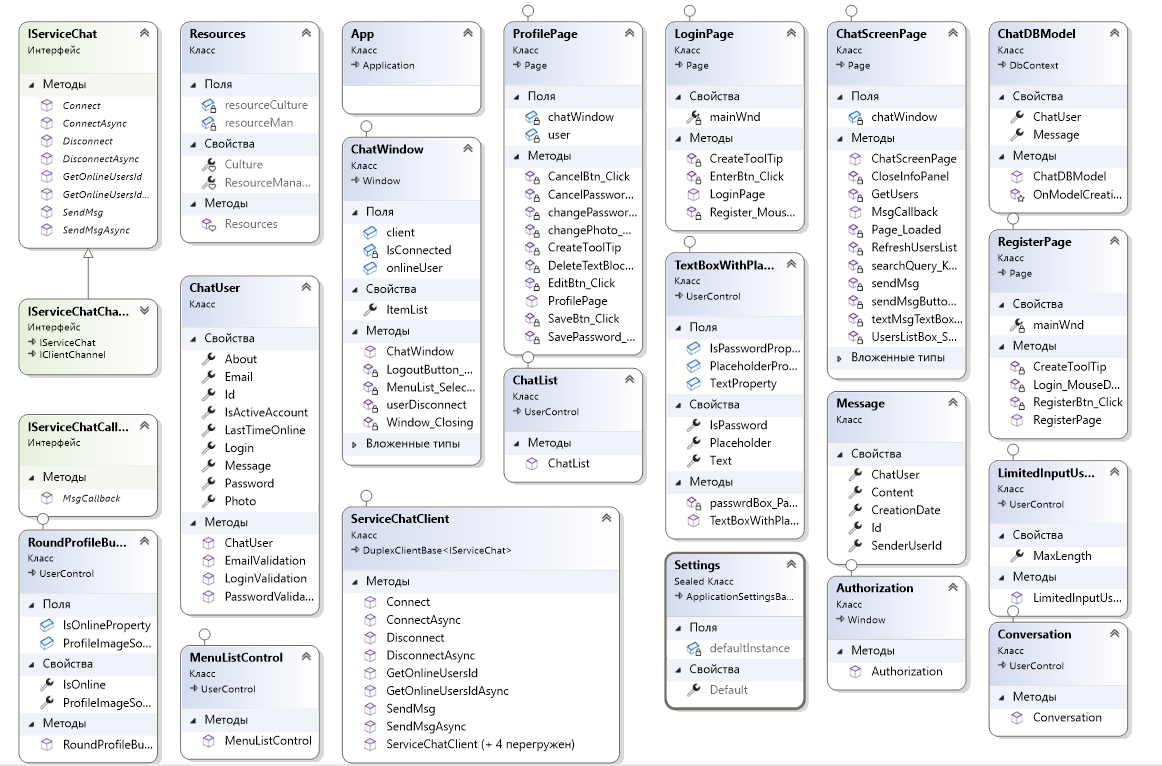
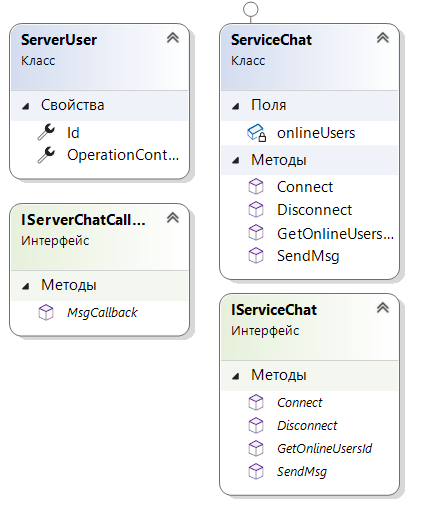


Диаграмма классов проекта WcfService.



ПРИЛОЖЕНИЕ Б

-- если БД Chat не существует, создаем её

if not exists (

select \* from sys.databases

where name = 'Chat')

begin

create database Chat;

end;

use Chat;

-- если таблицы не существует, создаем её

-- таблица с данными о пользователе

if not exists (

select \* from sys.objects

where object\_id = object\_id('ChatUser'))

begin

create table ChatUser(

Id bigint primary key identity(1, 1),

Login nvarchar(30) not null,

Email nvarchar(254) not null,

Password nvarchar(50) not null,

Photo varchar(max),

About nvarchar(2000),

LastTimeOnline smalldatetime,

IsActiveAccount bit not null,

);

end;

-- таблица с данными о каждом отдельном сообщении

if not exists (

select \* from sys.objects

where object\_id = object\_id('Message'))

begin

create table Message(

Id bigint primary key identity(1, 1),

Content nvarchar(3000) not null,

CreationDate smalldatetime not null,

SenderUserId bigint foreign key references ChatUser(Id),

);

end;

ПРИЛОЖЕНИЕ В

