МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

Факультет компьютерных наук

Кафедра обработки информации и машинного обучения

Курсовая работа по курсу

«Технологии программирования»

«Корпоративный мессенджер»

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_Р.Л. Гырбу,3 курс, д/о

Обучающийся\_\_\_\_\_\_\_\_\_\_\_\_ Д.Е. Семенихина,3 курс, д/о

Руководитель \_\_\_\_\_\_\_\_\_\_\_ М.С. Мишанин

Воронеж 2019

Содержание

[Введение 3](#_Toc10606870)

[1 Постановка задачи 4](#_Toc10606871)

[2 Анализ предметной области 6](#_Toc10606872)

[2.1 Глоссарий 6](#_Toc10606873)

[2.2 Аналоги решения задачи 6](#_Toc10606874)

[3 Проектирование приложения 10](#_Toc10606875)

[3.1 UML –диаграммы 10](#_Toc10606876)

[3.1.1 Диаграмма вариантов использования 10](#_Toc10606877)

[3.1.2 Диаграмма последовательности 12](#_Toc10606878)

[3.1.3 Диаграмма коммуникаций 14](#_Toc10606879)

[3.1.4 Диаграмма состояний 16](#_Toc10606880)

[3.1.5 Диаграмма деятельности 17](#_Toc10606881)

[3.1.6 Диаграмма развертывания 19](#_Toc10606882)

[4 Реализация приложения 20](#_Toc10606883)

[4.1 Средства реализации 20](#_Toc10606884)

[4.2 Интерфейс 20](#_Toc10606885)

[5 Тестирование приложения 23](#_Toc10606886)

[6 Оценка степени завершенности и перспективы доработки проекта 25](#_Toc10606887)

[Заключение 26](#_Toc10606888)

[Список литературы 27](#_Toc10606889)

## Введение

На сегодняшний день существует множество публичных сервисов для обмена информацией, но их использование зачастую не отвечает требованиям информационной безопасности. Компании постепенно начинают осознавать серьёзность угрозы. Утечка информации может стоить очень дорого, и не всегда это могут быть денежные потери, на карту может быть поставлена даже деловая репутация компании.

Более того, вы не можете быть уверены в том, что информация, которую отправляют ваши сотрудники, нигде не сохраняется и не протоколируется. Ведь, используя публичные сервисы, вы передаёте эту информацию через чужие серверы, которые никак не можете контролировать. И нет никакой гарантии, что при соответствующих обстоятельствах компания-владелец не вынуждена будет раскрыть эту информацию.

Отсюда вытекает актуальность работы. Именно поэтому в качестве направления деятельности была выбрана разработка приложения для защищенной передачи информации внутри компании.

Цель работы – реализовать приложение – корпоративный мессенджер с упором на соблюдение конфиденциальности и целостности передаваемых данных.

Предмет исследования – процесс коммуникации работников компании при помощи специальных приложений.

Объектом исследования – является приложение «Корпоративный мессенджер».

Практическая значимость состоит в возможности применения данного приложения для решения реальных задач, возникающих перед организациями, а именно: быстрый, удобный и безопасный процесс передачи информации между работниками компании.

# 1 Постановка задачи

**Цель:** разработать клиент - серверный корпоративный мессенджер с упором на соблюдение конфиденциальности и целостности передаваемых данных.

Для достижения цели были поставлены следующие задачи:

1. провести анализ предметной области;
2. провести проектирование приложения;
3. выбрать оптимальные средства разработки и реализовать приложение;
4. протестировать реализованное приложение;
5. рассмотреть возможности доработки и улучшения приложения.

**Требования:**

Приложение будет десктопным, то есть будет устанавливаться на компьютер каждого работника компании.

Данное приложение должно позволять пользователям общаться друг с другом, обмениваться файлами и изображениями.

Администратор может регистрировать нового пользователя в приложении и передавать ему данные для входа в корпоративный мессенджер (логин и пароль).

У зарегистрированного пользователя есть возможность осуществлять вход в приложение.

После входа в систему открывается страница, на которой отображаются диалоги со всеми зарегистрированными пользователями, из которых можно выбрать нужный диалог и начать обмен сообщениями, файлами, изображениями.

Из страницы с диалогами можно перейти в меню настроек. Зарегистрированный пользователь может изменять личные данные, такие как: фамилия, имя, пароль и адрес почты.  
Корпоративный мессенджер должен предоставлять удобный поиск сообщений.

Данная система направлена на пользование:

* физическими лицами внутри корпоративного сегмента.

# 2 Анализ предметной области

## 2.1 Глоссарий

**Корпоративный мессенджер** – это инструмент обмена мгновенными сообщениями, передачи файлов и других связанных функций для работы в компании.

**Компания** – коммерческое предприятие, торговое или промышленное объединение предпринимателей.

**Пользователь** – физическое лицо компании, которое будет использовать корпоративный мессенджер.

**Сообщение** – структурная единица обмена информацией между несколькими пользователями.

**Диалог** – обмен сообщениями между двумя пользователями.

**Чат** – канал для передачи сообщений между двумя пользователями.

**Администратор** – работник компании, ответственный за регистрацию новых пользователей в корпоративном мессенджере.

## 2.2 Аналоги решения задачи

Одним из корпоративных мессенджеров является iChat.

На рисунке 1 представлен интерфейс данного приложения:

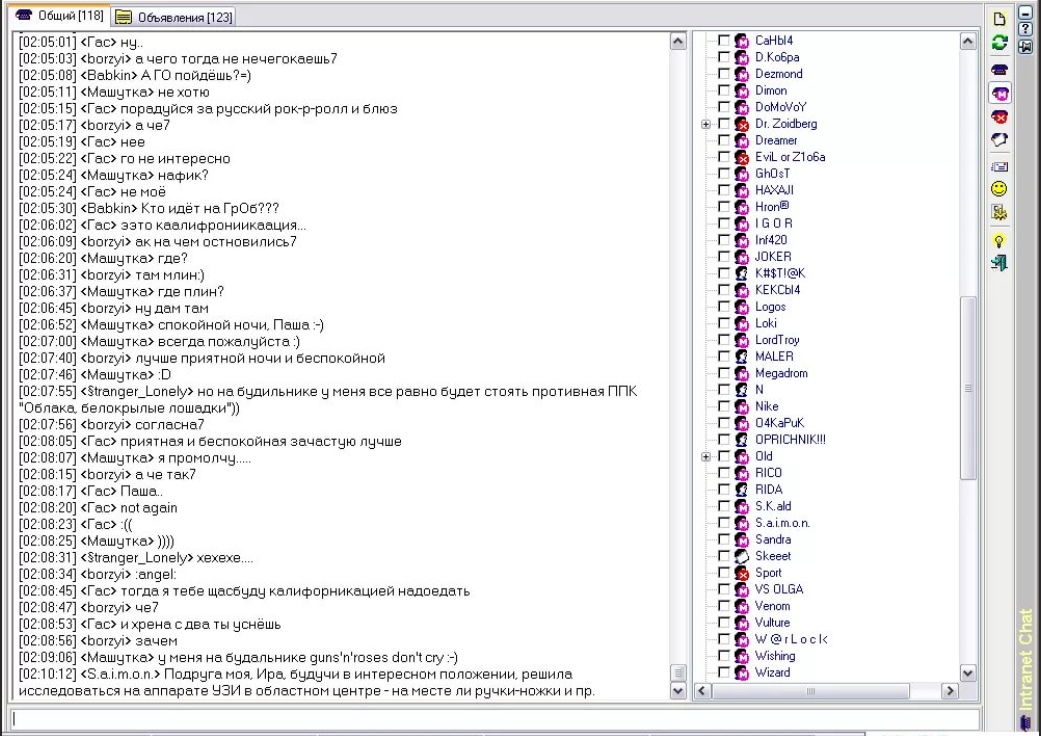


Рисунок – iChat

**Плюсы:**

* есть все необходимые функции — личные чаты и «доска объявлений»;
* всё бесплатно;
* можно поставить на свой сервер.

**Минусы:**

* нельзя передавать файлы;
* нельзя написать тому, кто сейчас офлайн;
* нет истории.

Следующее программное средство, которое используется для передачи информации внутри корпоративного сегмента: Rocket.Chat.

На рисунке 2 представлен интерфейс данного приложения:

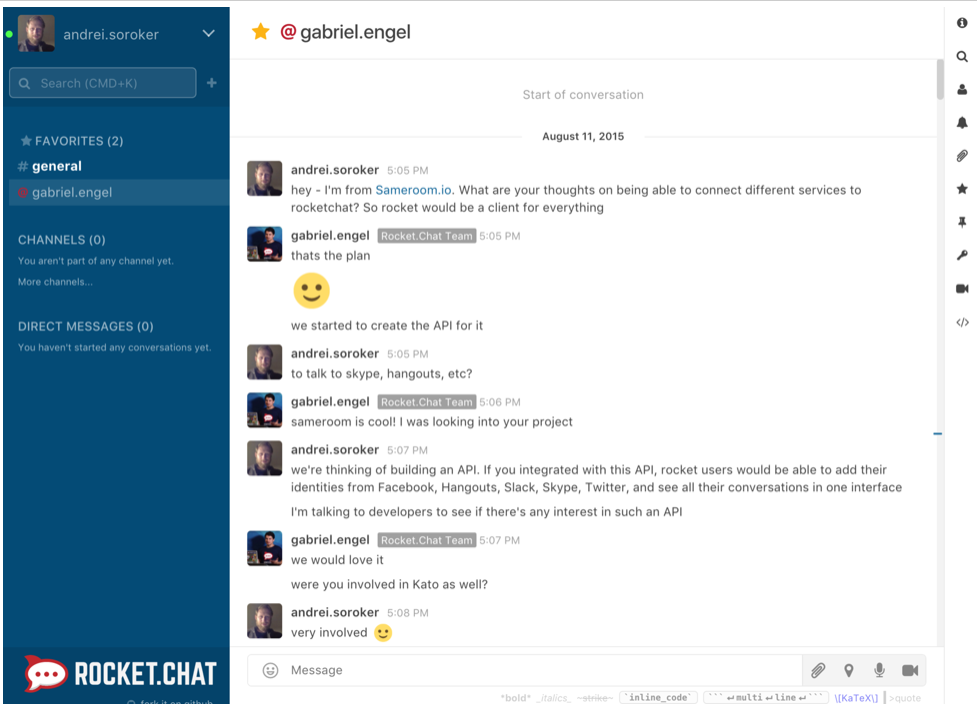


Рисунок - Rocket.Chat

**Плюсы:**

* есть файловый сервер;
* бесплатен;
* активно разрабатывается;
* отдельно можно настроить все уведомления.

**Минусы:**

* в интерфейсе нет списка всех пользователей;
* отсутствие передачи информации через HTTPS – протокол.

Таким образом, разрабатываемое приложение должно сохранить основные преимущества существующих решений и устранить ряд их недостатков.

А именно, разрабатываемый корпоративный мессенджер должен удовлетворять следующим свойствам:

* наличие списка всех пользователей;
* наличие передачи информации через HTTPS – протокол;
* возможность написать тому, кто сейчас офлайн;
* возможность передачи файлов;
* удобный и интуитивно понятный интерфейс.

# 3 Проектирование приложения

## 3.1 UML –диаграммы

Для проектирования приложения воспользуемся языком графического описания UML.

## 3.1.1 Диаграмма вариантов использования

В приложении будут два вида пользователей: «Администратор» и «Зарегистрированный пользователь».

Администратор отвечает за регистрацию новых пользователей в системе. После регистрации он дает новому пользователю информацию для осуществления авторизации: логин и пароль.

На рисунке 3 представлена диаграмма вариантов использования для Администратора.

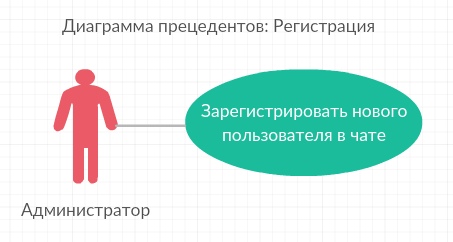
[](https://vk.com/photo133691292_456250636)

Рисунок - Диаграмма вариантов использования для Администратора

Зарегистрированный пользователь для входа в систему вводит на странице авторизации логин и пароль.

После того как пользователь осуществил вход в приложение, он может выполнять следующие действия:

* Выход из системы
* Выбор необходимого диалога с другим зарегистрированным пользователем
* Отправка сообщений
* Отправка файлов
* Отправка изображений
* Поиск по сообщениям
* Изменение персональных данных (фамилия, имя, email, пароль)

На рисунке 4 представлена диаграмма вариантов использования для зарегистрированного пользователя.

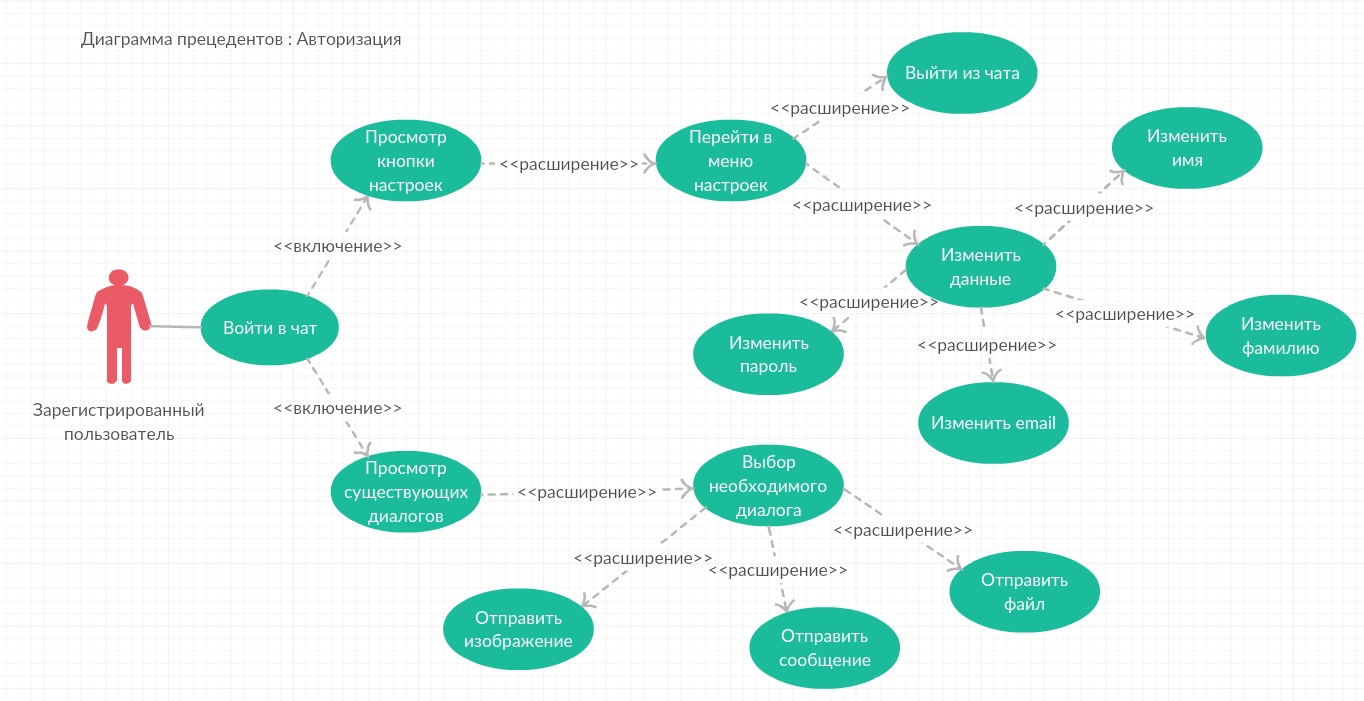
[](https://vk.com/photo133691292_456250635)

Рисунок - Диаграмма вариантов использования для Зарегистрированного пользователя

## 3.1.2 Диаграмма последовательности

На рисунке 5 представлена диаграмма последовательности для авторизации зарегистрированного пользователя в приложении.

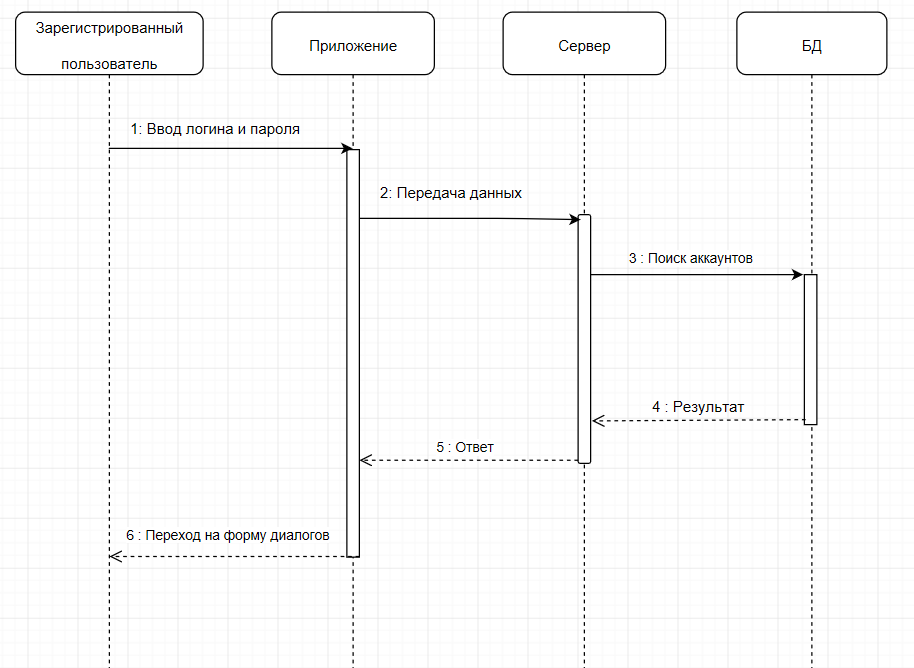


Рисунок - Диаграмма последовательности для авторизации

При запуске приложения пользователь попадает на страницу авторизации, на ней необходимо ввести логин и пароль. После ввода данные передаются на сервер, в свою очередь сервер отправляет запрос в базу данных, где происходит поиск аккаунтов.

Далее БД возвращает результат поиска на сервер, сервер дает ответ пользователю, если все данные были введены верно, открывается страница диалогов, на которой отображаются все пользователи системы и далее можно выполнять описанные в диаграмме вариантов использования для зарегистрированного пользователя действия.

На рисунке 6 представлена диаграмма последовательности для изменения персональных данных (фамилия, имя, email, пароль).

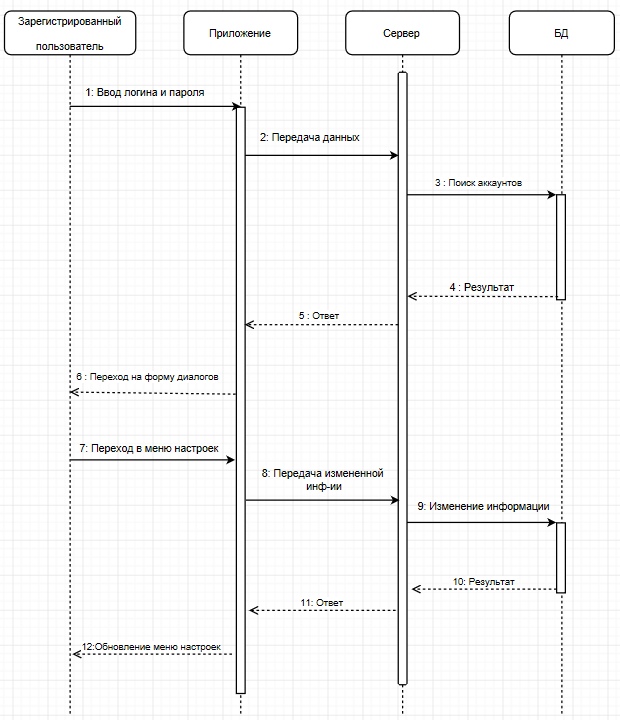
[](https://vk.com/photo133691292_456250894)

Рисунок - Диаграмма последовательности для изменения информации

Для того чтобы изменить персональные данные, пользователь для начала должен авторизоваться в системе, поэтому начало диаграммы последовательности для изменения информации повторяет диаграмму последовательности для авторизации. После успешной авторизации пользователь должен перейти в меню настроек и изменить необходимые данные. Измененные данные передаются на сервер, сервер передает их в базу данных, где происходит изменение информации. Далее база данных передает результат изменения данных на сервер, сервер дает ответ пользователю, происходит обновление меню настроек (обновляются измененные персональные данные).

## 3.1.3 Диаграмма коммуникаций

На рисунке 7 представлена диаграмма коммуникаций для авторизации пользователя.

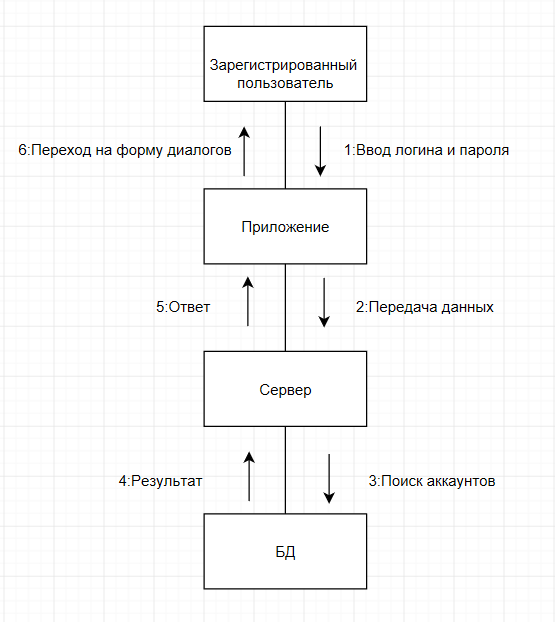


Рисунок - Диаграмма коммуникаций для авторизации

На рисунке 8 представлена диаграмма коммуникаций для изменения персональных данных.

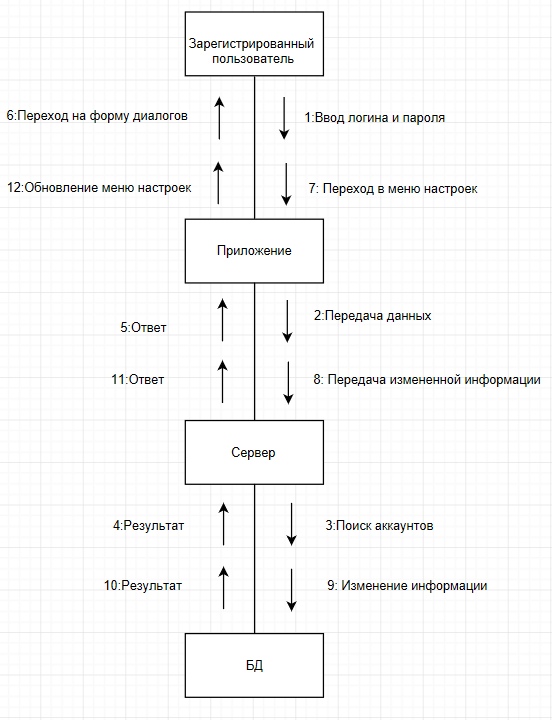
[](https://vk.com/photo133691292_456250895)

Рисунок - Диаграмма коммуникаций для изменения информации

Сообщения в диаграмме коммуникаций такие же, что и в диаграмме последовательности, поэтому их повторное описание не приводится.

## 3.1.4 Диаграмма состояний

На рисунке 9 представлена диаграмма состояний для входа в систему.

На рисунке 10 представлена диаграмма состояний для изменения данных.

Данные диаграмма изображена в упрощенной форме, более развернутые действия отображены на диаграммах деятельности.

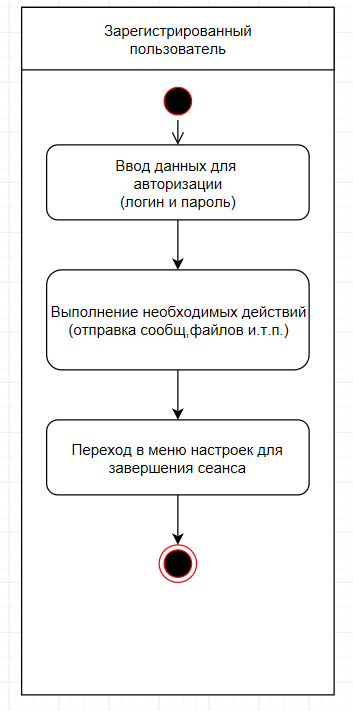


Рисунок - Диаграмма состояний для авторизации

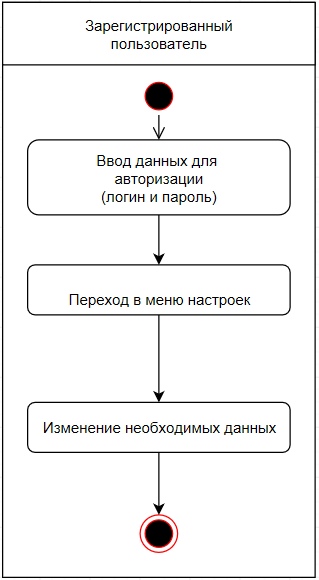
[](https://vk.com/photo133691292_456250896)

Рисунок - Диаграмма состояний для изменения информации

## 3.1.5 Диаграмма деятельности

Диаграммы деятельности являются расширениями диаграмм состояний, которые находятся в предыдущем разделе.

На рисунке 11 представлена диаграмма активности для авторизации пользователя. На данной диаграмме присутствуют четыре части: Пользователь, Приложение, Сервер, БД.

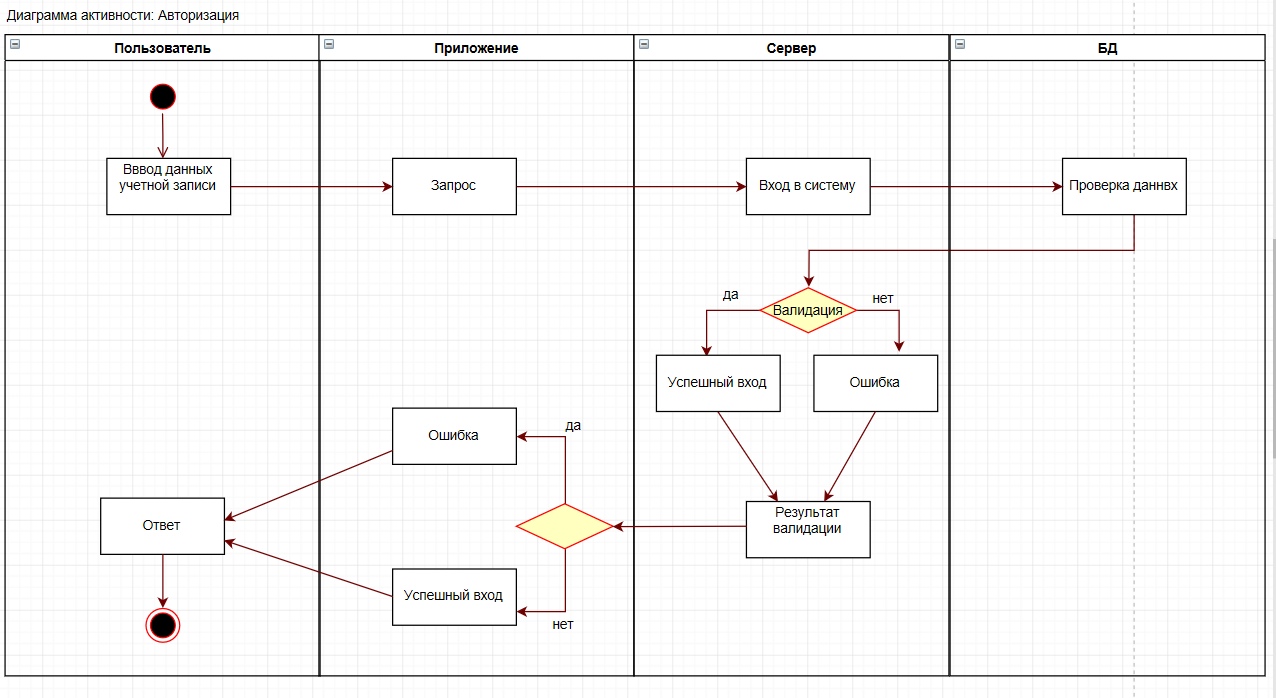
[](https://vk.com/photo133691292_456250625)

Рисунок 11 - Диаграмма активности для авторизации

На рисунке 12 представлена диаграмма активности для изменения персональных данных.

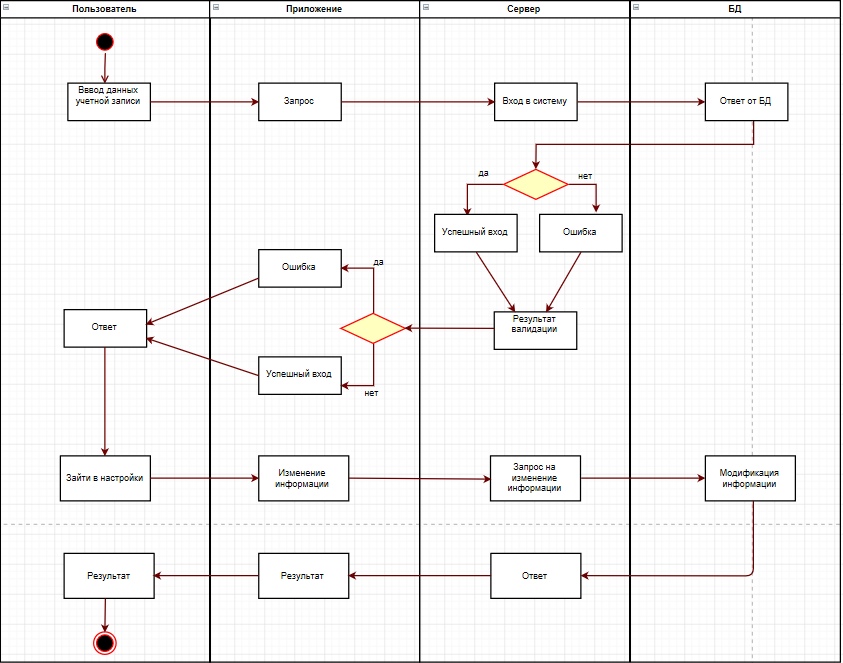
[](https://vk.com/photo133691292_456250892)

Рисунок 12 - Диаграмма активности для изменения персональных данных

## 3.1.6 Диаграмма развертывания

На рисунке 13 представлена диаграмма развертывания.

Она визуализирует элементы и компоненты программы, которые существуют на этапе её исполнения.

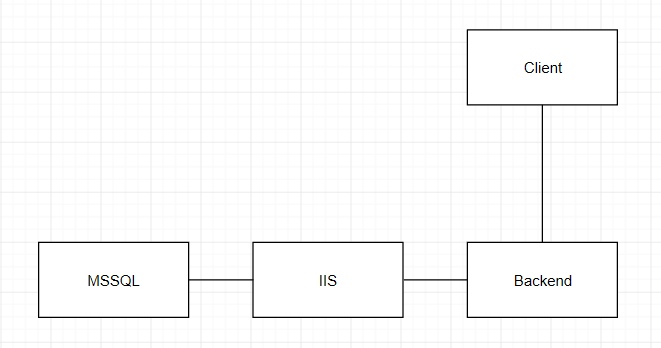
[](https://vk.com/photo133691292_456250897)

Рисунок 13 - Диаграмма развертывания

# 4 Реализация приложения

## 4.1 Средства реализации

Для реализации приложения были использованы следующие средства:

1. платформа .NET Core;
2. объектно – ориентированный язык программирования C#;
3. десктопное приложение WPF;
4. сервер Kestrel;
5. ORM – Entity Framework.

.NET Core является частью .NET Foundation, который существует для построения сообщества и внедрения инноваций в рамках развития фреймворка .NET Core. Использование этой платформы имеет много преимуществ: вы получаете больше свободы в контроле и изменениях проекта, прозрачность кода может снабдить вас информацией и послужить вдохновением для ваших собственных проектов на базе .NET Core.

Статус “открытости” также дает .NET Core большую устойчивость, поскольку в отличие от проприетарного программного обеспечения, которое часто бывает заброшено создателями, код, лежащий в основе инструментов этой платформы, всегда будет оставаться общедоступным.

DotNet Core – это кроссплатформенность. Эта уникальная программная платформа уже работает на Windows, Mac OS X и Linux. Хотя это может показаться странным шагом для Microsoft, но это важно в технологическом мире, который все больше ориентируется на гибкость и сегментированность, когда дело доходит до операционных систем и платформ.

В рамках кросс-совместимости платформа для разработки приложений включает в себя модульную инфраструктуру. Она выдается через NuGet, и вы можете получить доступ к пакетным функциям, а не к одной большой сборке. Как разработчик вы можете создавать легкие приложения, содержащие только необходимые пакеты NuGet, что сделает вашу программу безопаснее и производительнее.

Модульная инфраструктура также позволяет быстрее обновлять платформу .NET Core, поскольку затронутые модули могут обновляться и выпускаться по отдельности. Акцент на гибкость и быстрые релизы наряду с вышеупомянутым сотрудничеством позитивно позиционирует .NET Core в рамках движения DevOps.

Использование Entity Framework было принято после всего вышесказанного о dot net core. Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET 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. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

## 4.2 Frontend

Технология WPF является часть экосистемы платформы .NET и представляет собой подсистему для построения графических интерфейсов.

Если при создании традиционных приложений на основе WinForms за отрисовку элементов управления и графики отвечали такие части ОС Windows, как User32 и GDI+, то приложения WPF основаны на DirectX. В этом состоит ключевая особенность рендеринга графики в WPF: используя WPF, значительная часть работы по отрисовке графики, как простейших кнопочек, так и сложных 3D-моделей, ложиться на графический процессор на видеокарте, что также позволяет воспользоваться аппаратным ускорением графики.

Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML: вы можете создавать насыщенный графический интерфейс, используя или декларативное объявление интерфейса, или код на управляемых языках C# и VB.NET, либо совмещать и то, и другое.

Рисунок 14 – Схематическая архитектура WPF

Как видно на схеме, WPF разбивается на два уровня: managed API и unmanaged API (уровень интеграции с DirectX). Managed API (управляемый API-интерфейс) содержит код, исполняемый под управлением общеязыковой среды выполнения .NET - Common Language Runtime. Этот API описывает основной функционал платформы WPF и состоит из следующих компонентов:

* **PresentationFramework.dll**: содержит все основные реализации компонентов и элементов управления, которые можно использовать при построении графического интерфейса
* **PresentationCore.dll**: содержит все базовые типы для большинства классов из PresentationFramework.dll
* **WindowsBase.dll**: содержит ряд вспомогательных классов, которые применяются в WPF, но могут также использоваться и вне данной платформы

Листинг 1 – структура LoginPage.xaml

Листинг 2 -структура файла LoginPage.xaml.cs

## 4.3 Backend

В классе Startup делается следующее.

* Настраиваются все службы, необходимые приложению.
* Определяется конвейер обработки запросов.
* Код для настройки (или *регистрации*) служб добавляется в метод Startup.ConfigureServices. *Службы* — это компоненты, которые используются в приложении. Например, объект контекста Entity Framework Core является службой.
* Код для настройки конвейера обработки запросов добавляется в метод Startup.Configure. Конвейер состоит из ряда компонентов *ПО промежуточного слоя*. Например, это ПО может обрабатывать запросы для статических файлов или перенаправлять HTTP-запросы в HTTPS. Каждый компонент ПО промежуточного слоя выполняет асинхронные операции в HttpContext, а затем либо вызывает следующий компонент в конвейере, либо завершает запрос.

Kestrel — это кроссплатформенный веб-сервер. Он часто запускается в реконфигурации прокси-сервера с помощью IIS. В ASP.NET Core 2.0 и более поздних версиях можно использовать Kestrel в роли общедоступного пограничного сервера с прямым доступом через Интернет.

Листинг 3 – фрагмент кода Startup.cs

## 4.4 Интерфейс

При открытии приложения пользователь попадает на главную страницу.

На рисунке 14 представлена главная страница приложения.

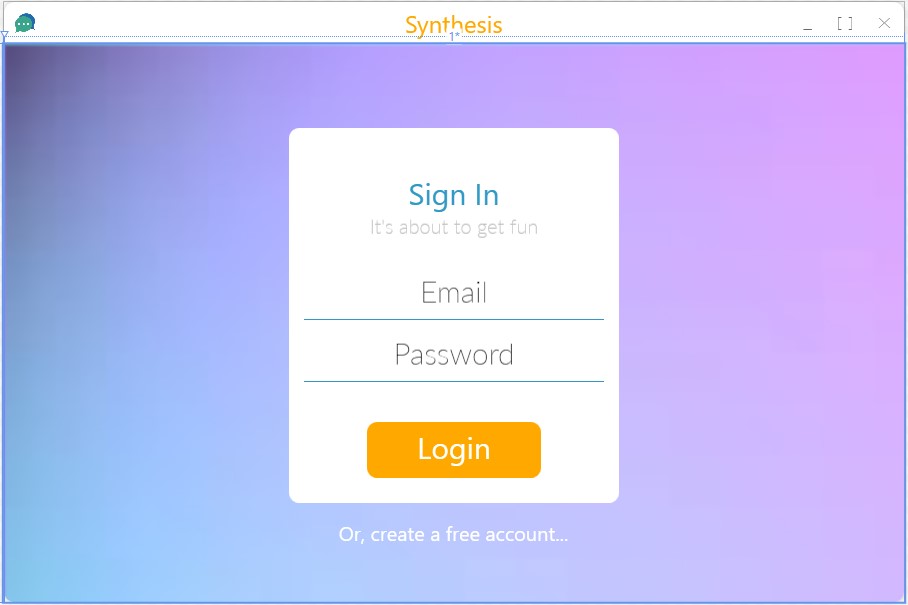


Рисунок - Страница авторизации

Далее должен быть введен Email и Password для авторизации в системе, если пользователь не зарегистрирован есть возможность создать новый аккаунт (регистрация проводится специальным человеком – Администратором, после чего новый работник корпоративного сегмента получает данные для входа в систему).

На рисунке 15 представлена страница регистрации.

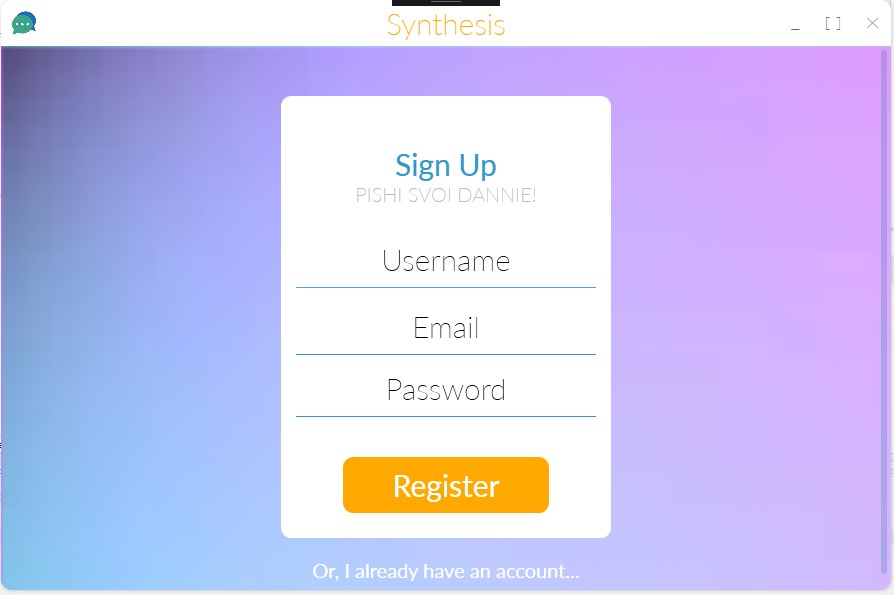


Рисунок 15 - Страница регистрации

На рисунке 16 представлен вид страницы диалогов, на которую пользователь попадает после авторизации.

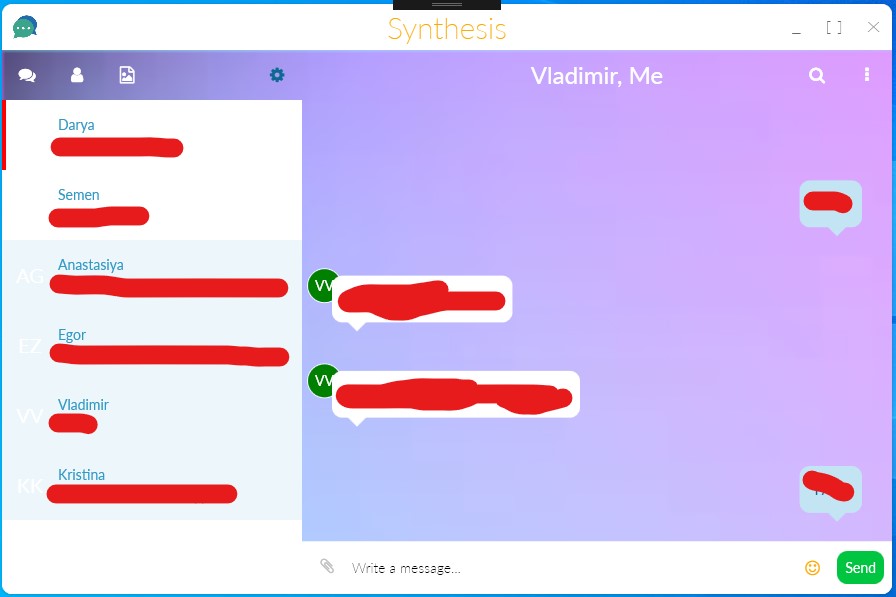


Рисунок 16 - Страница диалогов

На этой странице авторизованный пользователь может выполнить следующие действия:

* выбор необходимого диалога с другим зарегистрированным; пользователем
* отправка сообщений;
* отправка файлов;
* отправка изображений;
* поиск по сообщениям.

Также можно зайти в меню настроек. В нем пользователю доступны следующие действия:

* выход из системы;
* изменение персональных данных (фамилия, имя, email, пароль).

На рисунке 17 представлено меню настроек:

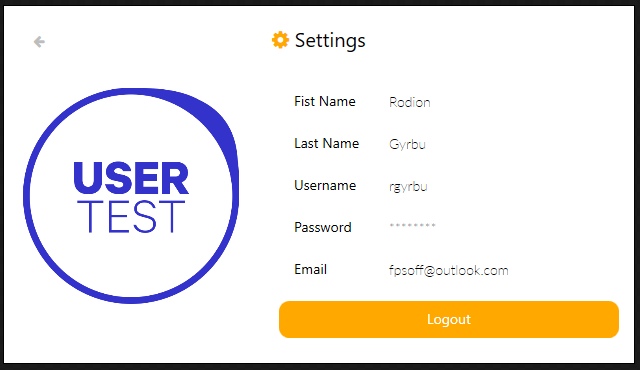
[](https://vk.com/photo260584838_456244490)

Рисунок 17 - Меню настроек

# 5 Тестирование приложения

Тестирование было проведено в соответствии с документом «План тестирования», который прилагается к данной курсовой работе.

Основные виды тестирования, которые были проведены:

1. Ручное тестирование – тестовые циклы были произведены вручную, с последующей ручной фиксацией результатов по каждому тесту.

2. Smoke testing – поверхностное тестирование всех модулей приложения на предмет их работоспособности. Тестирование проводилось при каждом изменении версии проекта. Была проверена возможность осуществления основных функций, а именно: авторизация, выбор диалога, отправка сообщения, переход в меню настроек, изменение персональных данных, выход из системы.

3. Sanity testing – тестирование работы конкретных функций системы для доказательства того, что система работает согласно требованиям в Техническом Задании. В рамках данного теста были проверены возможности отправки сообщений, файлов, изображений.

4. Usability testing – тестирование того, насколько легко конечный пользователь сможет освоить систему. Здесь проверялось наличие сообщений-подсказок для пользователя, при неправильных действиях с его стороны.

5. Negative testing – тестирование поведении системы, при попытке ввода некорректных данных. Было протестировано поведение системы, при попытке ввода в соответствующие поля некорректных символов (например, русские буквы) при регистрации в системе.

6. E2E testing – тестирование системы от начала до конца так, как с ней должен работать пользователь.

Следует отметить, что приложение десктопное, поэтому необходимо провести ряд следующих тестов:

* тестирование инсталляции;
* тестирование обновления;
* тестирование деинсталляции.

Выполняя **тестирование установки** проверяется:

1. Запускается ли программа после установки
2. Расположение программы в файловой системе по умолчанию
3. Расположение программы в файловой системе, если путь сохранения изменен пользователем
4. Наличие ярлыков на рабочем столе
5. Есть ли установленный компонент в меню Пуск  > Программы
6. При установке обратить внимание на издателя
7. Установка программы для текущего пользователя/для всех пользователей компьютера
8. Установка пользователем с правами админа
9. Установка пользователем без прав админа

Выполняя **тестирование удаления** проверяем:

1. Файлы должны удалиться
2. Ярлык с рабочего стола исчез
3. Удалена ли запись из меню Пуск> Все программы

# 6 Оценка степени завершенности и перспективы доработки проекта

Данный проект находится в рабочем состоянии. Ошибок и сбоев при тестировании и работе не выявлено.

Задачи, которые были описаны в ТЗ выполняются, а именно:

1. зарегистрированный пользователь может осуществлять следующие действия:

* вход в систему;
* выход из системы;
* выбор необходимого диалога с другим пользователем;
* отправка файлов;
* отправка изображений;
* поиск по сообщения;
* изменение персональных данных.

1. администратор может выполнять следующие действия:

* зарегистрировать нового пользователя.

Приложение осуществляет безопасную передачу информации, сохраняя её целостность.

Однако приложение «Корпоратвный мессенджер» выполняет только основные функции, которые требуются для передачи информации внутри компании, поэтому в дальнейшем его можно расширять (добавить возможность отправки голосовых сообщений и так далее).

Также можно детально проработать интерфейс приложения (к примеру, показывать статус онлайн\офлайн, добавить возможность отправлять смайлики).

## Заключение

В ходе выполнения курсовой работы были достигнуты следующие результаты:

1. проведен анализ предметной области;
2. произведено проектирование приложения;
3. выбраны оптимальные средства разработки и реализовано приложение;
4. протестировано реализованное приложение;
5. рассмотрены возможности доработки и улучшения приложения.

## Список литературы

1. <https://habr.com/ru/company/edison/blog/267569/>
2. <https://qaevolution.ru/625-2/>
3. <https://habr.com/ru/post/276593/>
4. <http://michaelsmirnov.blogspot.com/2011/03/uml.html>