

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра информационные системы и технологии

Разработка веб-мессенджера «Linking»

Курсовой проект

09.03.02 Информационные системы и технологии
Программная инженерия в информационных системах

Обучающийся _____ М.Д. Шеменев, 3 курс, д/о

Обучающийся _____ И.В. Ставер, 3 курс, д/о

Обучающийся _____ М.Р. Попов, 3 курс, д/о

Руководитель _____ В.С. Тарасов, ст. преподаватель _____.20__

Воронеж 2023

Содержание

Введение	4
1 Постановка задачи	5
1.1 Требования к разрабатываемой системе	5
2 Анализ предметной области	6
2.1 Терминология	6
2.2 Обзор аналогов	7
2.2.1 Telegram.....	7
2.2.2 WhatsApp	9
2.3 Требования к функции	11
2.3.1 Приветственная страница.....	11
2.3.2 Главная страница	11
2.3.3 Выпадающий список поиска	13
2.3.4 Страница чата.....	15
2.3.5 Страница канала.....	15
2.3.6 Страница настроек канала	17
2.3.7 Всплывающее окно добавления ролей	18
2.3.8 Всплывающее окно создания канала	19
2.3.9 Страница аккаунта пользователя	20
2.3.10 Страница изменения пароля.....	20
2.3.11 Страница изменения почты	21
2.3.12 Страница входа в аккаунт.....	22
2.3.13 Страница регистрации	23
2.3.14 Страница восстановление пароля	24

3 Диаграммы	26
3.1 Диаграмма прецедентов	26
3.2 Диаграмма классов	27
3.3 Диаграмма последовательности	28
3.4 Диаграмма активности	31
3.5 Диаграмма развёртывания	34
3.6 Диаграмма сотрудничества	35
3.7 Диаграмма объектов	35
3.8 Диаграмма состояний	36
3.9 ER-Диаграмма	39
3.10 Физическая схема базы данных	40
3.11 IDEF0 Диаграмма	41
4 Реализация	43
4.1 Средства реализации	43
4.2 Реализация Backend	46
4.3 Реализация Frontend	49
4.4 Тестирование	52
4.4.1 Модульное тестирование	52
4.4.2 Дымовое тестирование	53
4.4.3 GUI-тестирование	55
Заключение	59
Список использованных источников	60

Введение

В современном обществе очень большую роль играют социальные сети и разнообразные мессенджеры как средства виртуального общения. Растет число людей, которые могут часами проводить в виртуальном общении, а многим оно заменяет реальную коммуникацию. Также мессенджеры могут использоваться не только в повседневной жизни, но и в профессиональной деятельности. В условиях необходимости быстро передать информацию, задать вопрос и получить на него ответ, мессенджеры для профессионального общения становятся крайне необходимы.

Мессенджеры – это новый способ контактирования между людьми, вне зависимости от географического положения, посредством обмена мгновенными сообщениями. Потому и растет актуальность среди приложений, предоставляемых возможности приватного взаимодействия. Именно этим обусловлена актуальность данной работы.

Целью курсовой работы является разработка веб-приложения мессенджера для обеспечения удобной коммуникации пользователей.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить и описать имеющиеся наиболее популярные мессенджеры, выявить их достоинства и недостатки;
2. Определить требования к веб-приложению мессенджера;
3. Разработать мессенджер с визуальным интерфейсом;
4. Приступить к реализации мессенджера.

К методам курсовой работы относятся: системный анализ, моделирование СУБД, информационное моделирование данных. Веб-приложение реализовано на языке Java с использованием SQL-запросов. В коде применялись стандарты объектно-ориентированного подхода к программированию.

1 Постановка задачи

1.1 Требования к разрабатываемой системе

Целью данной курсовой работы является создание веб-приложения мессенджера, которое будет удовлетворять следующим требованиям:

1. Кастомизация ролей – создание разнообразных ролей для канала;
2. Возможность неавторизованного пользователя просматривать историю сообщений каналов;
3. Возможность обмениваться сообщениями двум или группе пользователей;
4. Возможность добавить важные сообщения в раздел избранных в один клик.

Для реализации цели были поставлены следующие задачи:

1. Проанализировать и произвести выбор технологий реализации и необходимых программных платформ;
2. Изучить и описать имеющиеся наиболее популярные мессенджеры, выявить их достоинства и недостатки;
3. Определить требования к веб-приложению мессенджера;
4. Разработать мессенджер с визуальным интерфейсом;
5. В соответствии с техническим заданием провести разработку мессенджера.

2 Анализ предметной области

2.1 Терминология

Мессенджер – приложение для общения.

Сервер – выделенный или специализированный компьютер для выполнения сервисного программного обеспечения.

База данных – это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД).

HTTP – это протокол, позволяющий получать различные ресурсы, например, HTML-документы. Протокол HTTP лежит в основе обмена данными в Интернете.

SQL запросы – это наборы команд для работы с реляционными базами данных.

Дизайн-макет – это схематичное изображение финальной идеи с указанием всех деталей. В нем указываются концепция, шрифты, тексты, изображения, расположение всех элементов и общая картина продукта.

Аутентификация – процедура проверки подлинности. Например, проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе данных.

Авторизация – предоставление определенному лицу или группе лиц прав на выполнение определенных действий.

Фреймворк – программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

SQL-инъекция – внедрении в запрос произвольного SQL-кода, который может повредить данные, хранящиеся в БД или предоставить доступ к ним.

HTTPS – расширение протокола HTTP для поддержки шифрования в целях повышения безопасности.

Пользователь – человек, который использует приложение.

Аккаунт или учетная запись – это персональная страница пользователя или личный кабинет, который создается после регистрации на сайте.

Frontend – клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса.

Backend – программно-аппаратная часть сервиса, отвечающая за функционирование его внутренней части.

REST – архитектурный стиль взаимодействия компонентов распределённого приложения в сети. [1]

2.2 Обзор аналогов

2.2.1 Telegram

Telegram – это бесплатный мессенджер для мгновенного обмена аудио- и видеосообщениями, изображениями, GIF, стикерами, документами в разных форматах (XLS, PDF, DOCX и прочих).

Возможности приложения:

- обмен файлами с одним или несколькими пользователями из любой точки мира;
- бесплатные аудио- и видеозвонки одному или нескольким абонентам;
- создание групповых чатов с возможностью добавления до 200 пользователей;
- создание групп численностью до 10 000 участников;
- создание каналов и распространение контента;
- просмотр и поиск каналов по ключевым словам;
- создание секретных чатов, данные которых «сгорают» через 24 часа;
- создание, загрузка и отправка стикеров;

- хранение любого объема данных в облачном хранилище (в «Избранном»);
- создание и использование автоматизированных ботов;
- редактирование фото перед отправкой;
- группировка чатов и каналов.

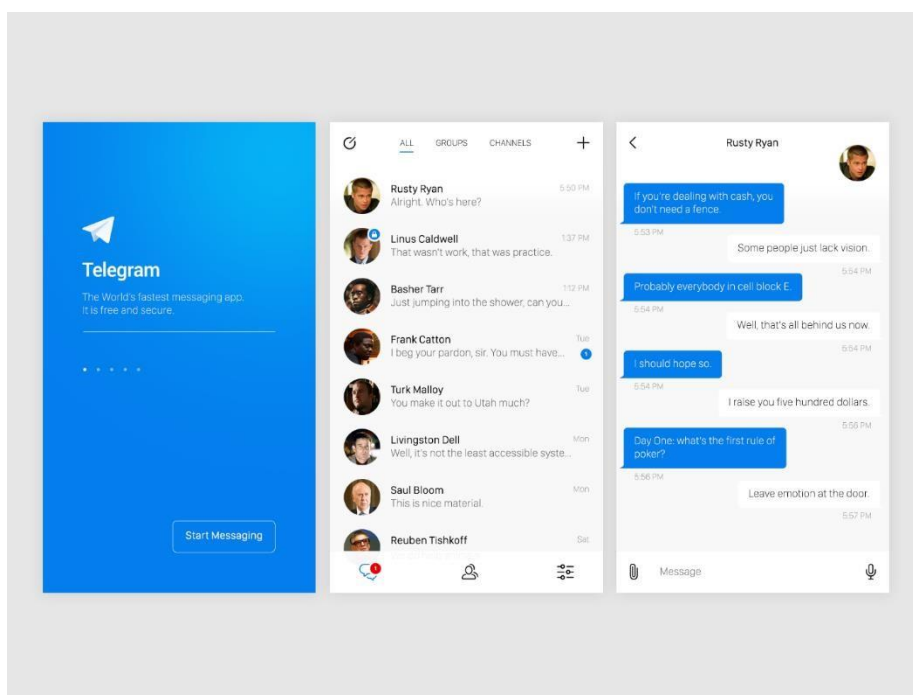


Рисунок 1 - Интерфейс страницы «Telegram»

Недостатки:

- Сбои в работе платформы. Пользователи иногда сталкиваются с техническими ошибками, из-за которых платформа на некоторое время перестает работать;
- Реклама спонсоров Telegram в пользовательских каналах. В октябре 2021 года в мессенджере запустили автоматическое размещение рекламы партнеров;
- Спам. Чтобы написать сообщение пользователю в Telegram, достаточно знать его никнейм. Когда вы активно пользуетесь мессенджером (вступаете в чаты, оставляете комментарии в

обсуждениях), кто-то может сохранить ваши видимые данные, чтобы в будущем делать спам-рассылки;

— Риск мошенничества. За счет полной анонимности злоумышленники почти всегда избегают наказания. Они выманивают у пользователей деньги и персональную информацию, предлагают вложиться в фейковые проекты, что ставит под сомнение вопрос безопасности площадки.

2.2.2 WhatsApp

WhatsApp – один из популярных мессенджеров. Приложение полностью бесплатно и работает на любых платформах. Есть также Web-версия программы, которая запускается прямо в окне браузера. Сегодня аудитория WhatsApp составляет около полутора миллиардов человек. Количество сообщений, отправляемых ежедневно, превышает 50 млрд. Работает приложение через интернет, поэтому с современным телефоном и установленным мессенджером вы всегда можете быть на связи: дома, в офисе или в автомобиле.

Его основные функции и возможности:

- отправка текстовых сообщений любому пользователю приложения;
- создание голосовых любой длительности;
- совершение звонков в любую точку мира;
- отправка фотографий и документов любого формата;
- организация коллективного общения при помощи текстовых сообщений и видеозвонков;
- сохранение важной информации, изображений, документов и видео в память устройства.

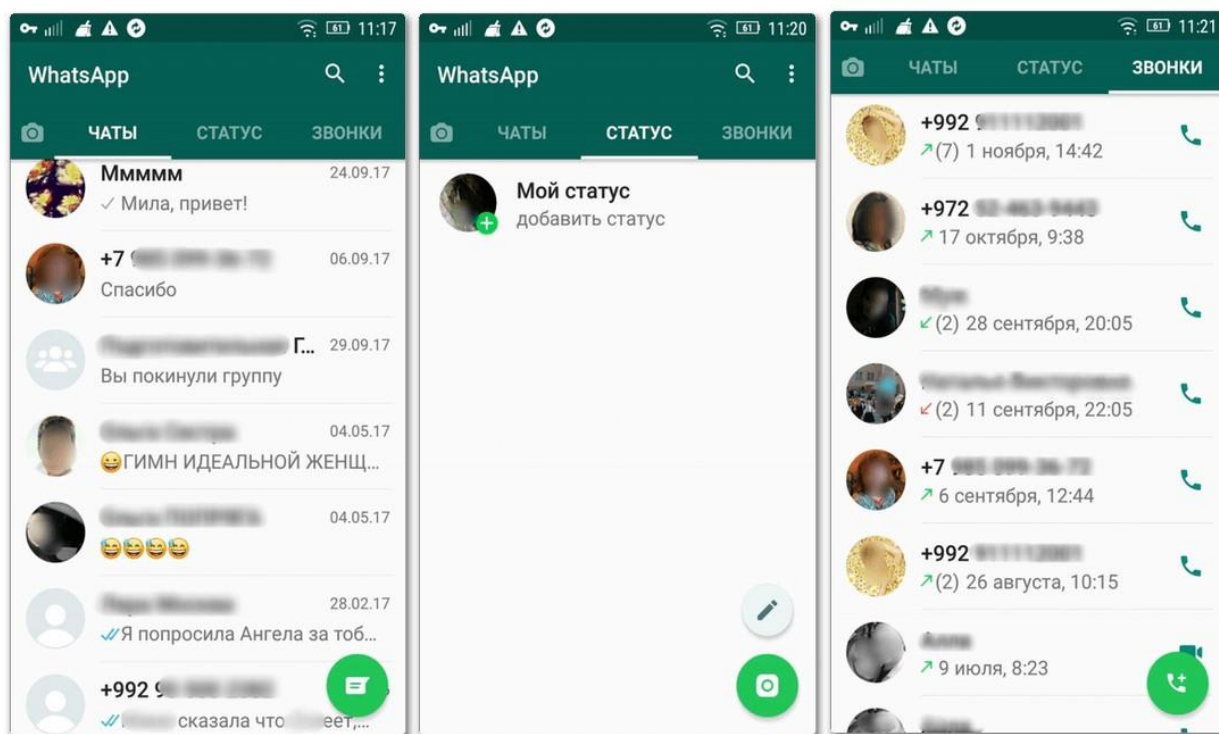


Рисунок 2 - Интерфейс страницы «WhatsApp»

Недостатки:

- Отсутствие возможности отменить отправку сообщения. Вы можете удалить их, но это не гарантирует вам того, что его не прочитают, ведь существует множество способов, позволяющих читать удалённые сообщения в WhatsApp;
- Отсутствие возможности скрыть номер телефона во время обмена контактами для общения в WhatsApp;
- Может занимать много места в памяти из-за накопления мультимедиа файлов, которыми вы обмениваетесь со своими контактами;
- Большое количество спама;
- Новые правила использования и обновление политики;
- конфиденциальности, в соответствии с которыми WhatsApp будет обмениваться пользовательскими данными с Facebook.

2.3 Требования к функции

2.3.1 Приветственная страница

На данном экране отображен логотип приложения и приветственное сообщение.

Снизу расположен макет страницы. При нажатии на кнопку «Начать» пользователь переходит на главную страницу неавторизованного пользователя.



Рисунок 3 – Приветственная страница

2.3.2 Главная страница

Пользователь (авторизованный и неавторизованный) имеет возможность перейти на главную страницу.

Для неавторизованного пользователя на этой странице отобразится интерфейс главной страницы с ограниченным функционалом, позволяющим искать чаты/каналы.

В случае авторизованного пользователя, ему будет показан интерфейс главной страницы с полным функционалом, позволяющим перейти в чат или канал из списка, просматривать избранные сообщения, редактировать аккаунт пользователя, создавать новый канал для группового общения, выйти из аккаунта.

Снизу расположены макеты страниц с кнопками. При нажатии на соответствующую кнопку пользователь переходит на страницы: чат/канал, аккаунт пользователя, создать канал, входа.

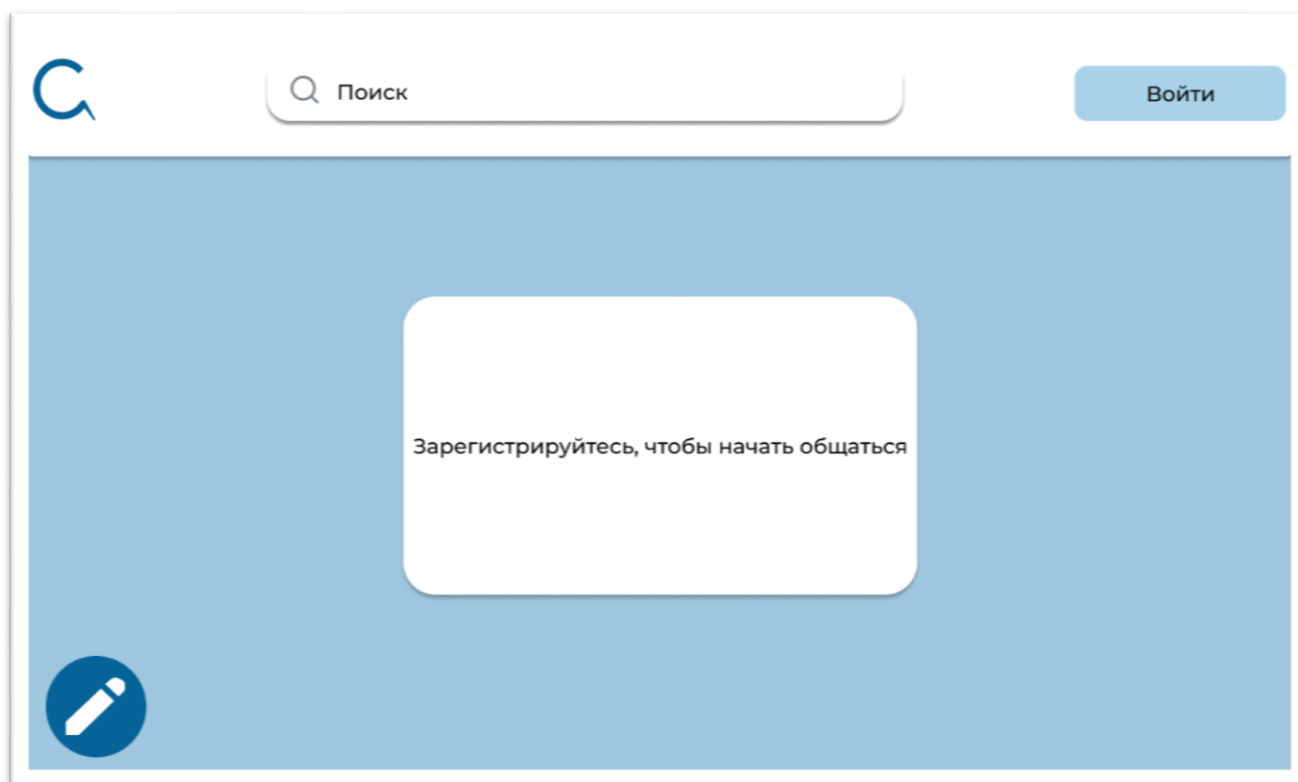


Рисунок 4 – Главная страница для неавторизованного пользователя

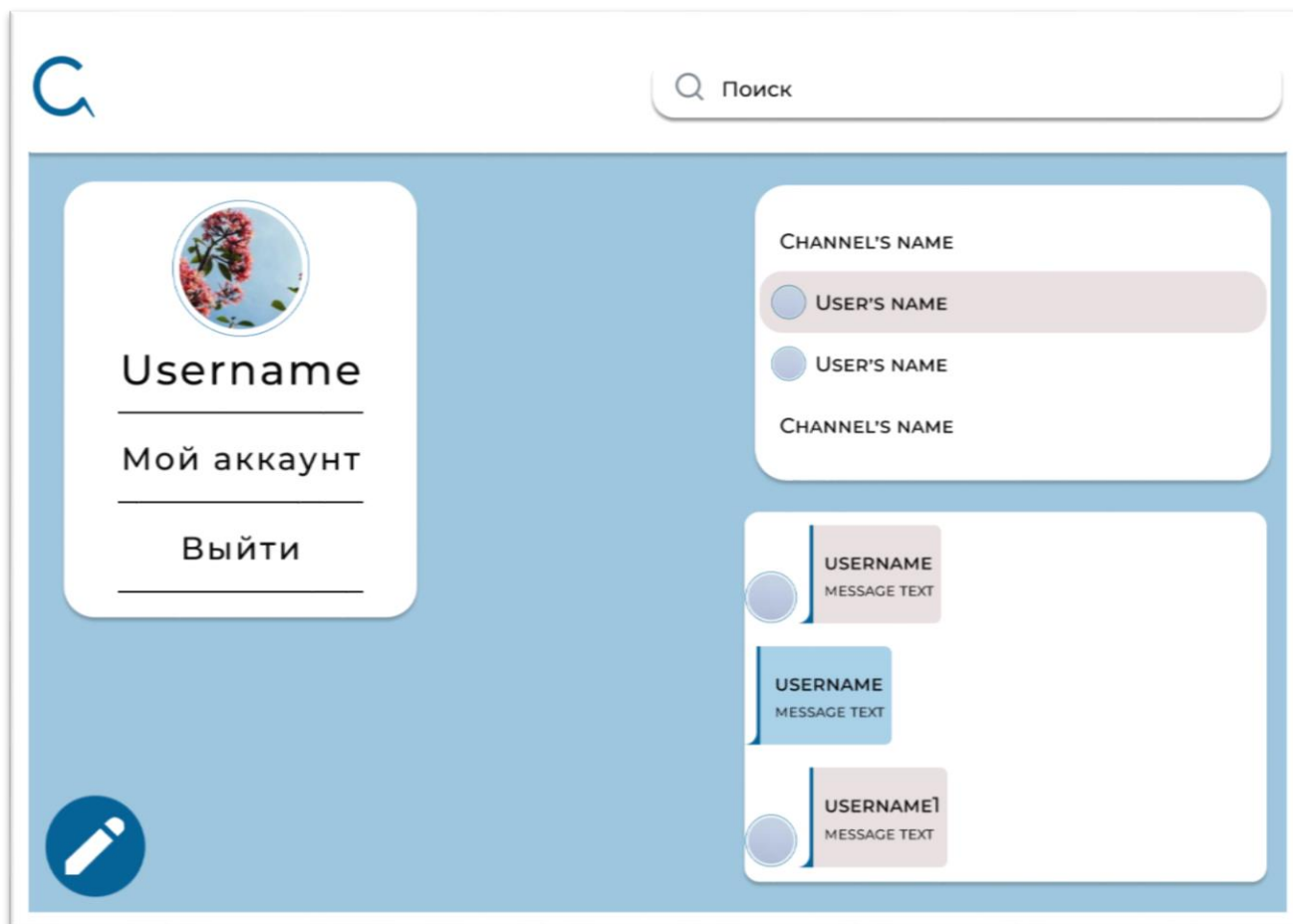


Рисунок 5 – Главная страница для авторизованного пользователя

2.3.3 Выпадающий список поиска

При вводе данных в поле поиска пользователь имеет возможность перейти на страницу, где отобразится история сообщений.

Снизу расположены главные страницы для неавторизованного и авторизованного пользователей с выпадающим списком поиска, где при нажатии на соответствующий элемент списка пользователь переходит на страницы канала или чата.

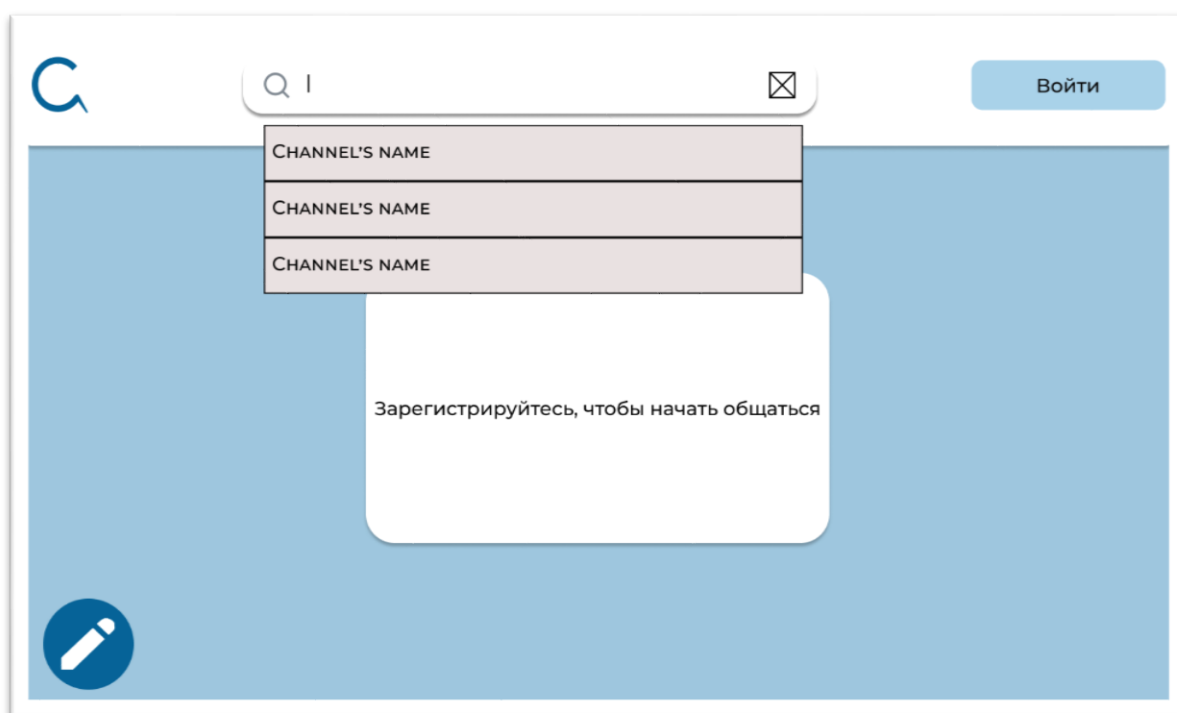


Рисунок 6 – Выпадающий список пользователя для неавторизованного пользователя



Рисунок 7 – Выпадающий список пользователя для авторизованного пользователя

2.3.4 Страница чата

Пользователь (авторизованный) имеет возможность при выборе чата из списка на главной странице или из выпадающего списка поиска перейти на его страницу для общения с другим зарегистрированным пользователем. Снизу расположен макет страницы с кнопками. При нажатии на соответствующую кнопку пользователь может перейти на главную страницу или отправить сообщение.

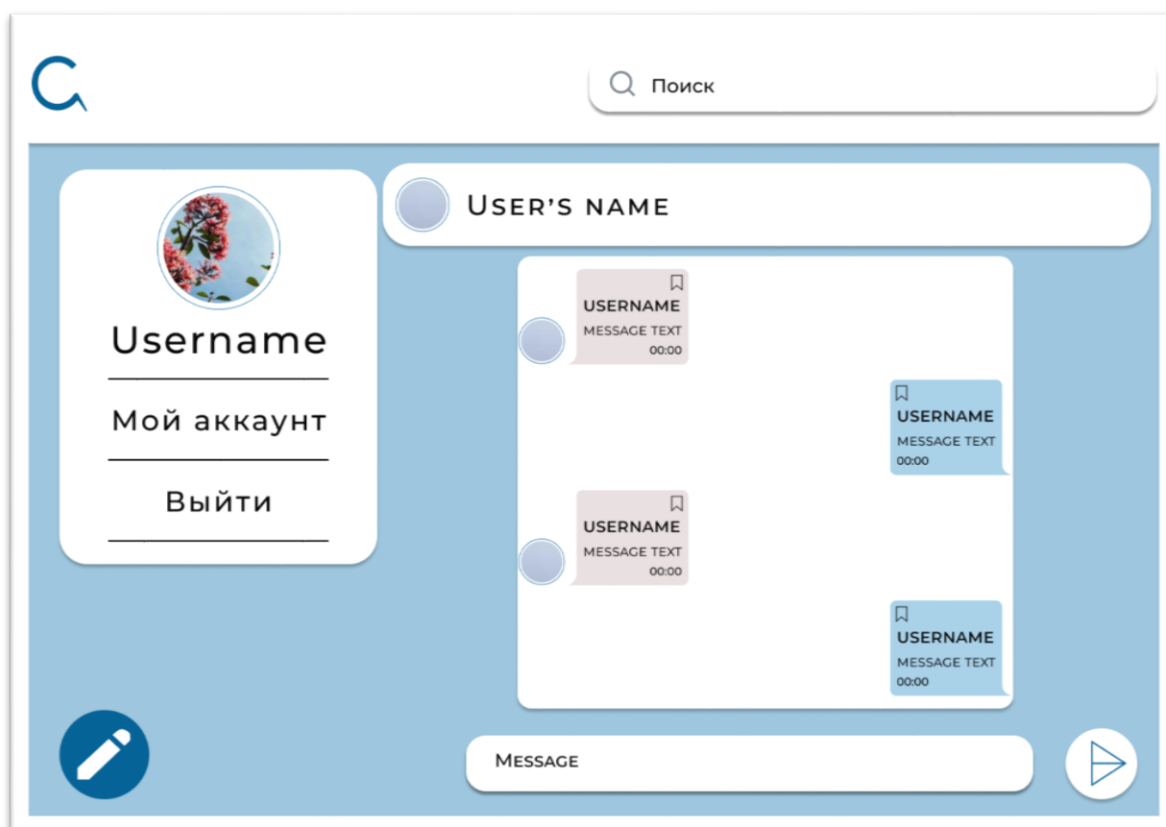


Рисунок 8 – Страница чата

2.3.5 Страница канала

Пользователь (авторизованный) имеет возможность при выборе канала из списка на главной странице или из выпадающего списка поиска перейти на его страницу для общения между группой пользователей.

Пользователь (неавторизованный) имеет возможность при выборе канала из выпадающего списка поиска перейти на его страницу для просмотра

истории сообщений. При нажатии на кнопку «Присоединиться» неавторизованный пользователь переходит на страницу входа в аккаунт.

Для авторизованного пользователя с правами участника доступны кнопки: «Присоединиться» / «Покинуть», отправить сообщение.

Для авторизованного пользователя с правами администратора доступны кнопки: «Присоединиться» / «Покинуть», отправить сообщение, настройки канала.

Для авторизованного пользователя с правами создателя доступны кнопки: отправить сообщение, настройки канала.

Снизу расположены макеты страниц с кнопками. При нажатии на соответствующую кнопку пользователь может перейти на главную страницу, открыть настройки канала, вступить или выйти из канала и отправить сообщение.

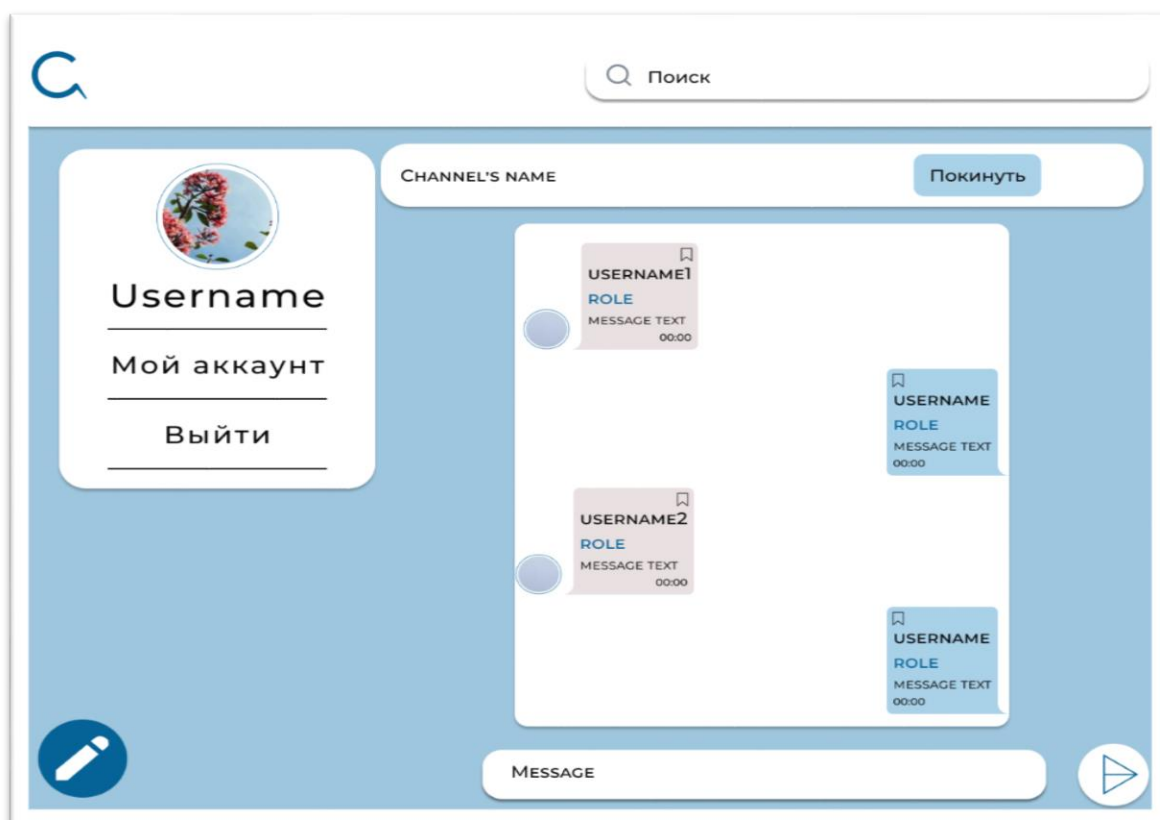


Рисунок 9 – Страница канала для пользователя с правами участника

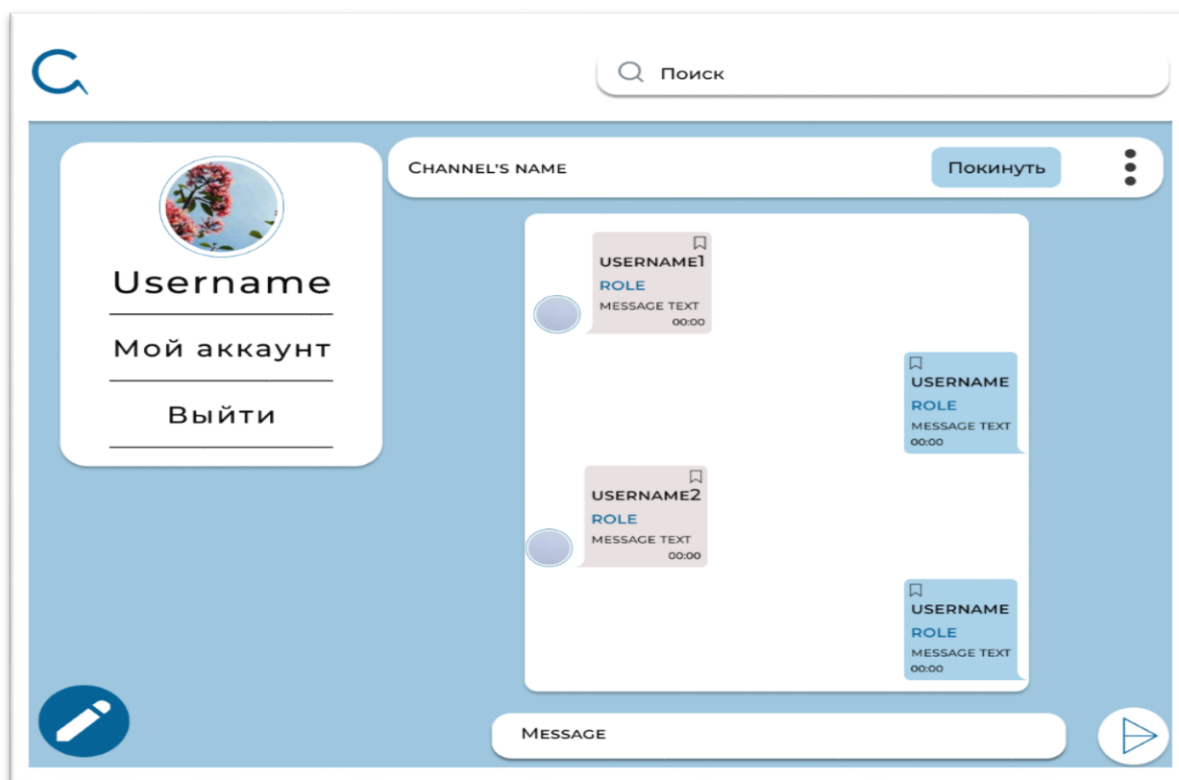


Рисунок 10 – Страница канала для пользователя с правами администратора и создателя

2.3.6 Страница настроек канала

Пользователь (авторизованный, с правами администратора или создателя) имеет возможность перейти на страницу настроек канала.

Создатель имеет право изменять имя канала, добавить роль, удалить любого пользователя, удалить канал. Администратор не может удалить канал или другого администратора.

Снизу расположен макет страницы с кнопками. При нажатии на соответствующую кнопку пользователь может: вернуться на страницу канала, удалить пользователя, добавить роль, удалить канал, изменить имя канала.

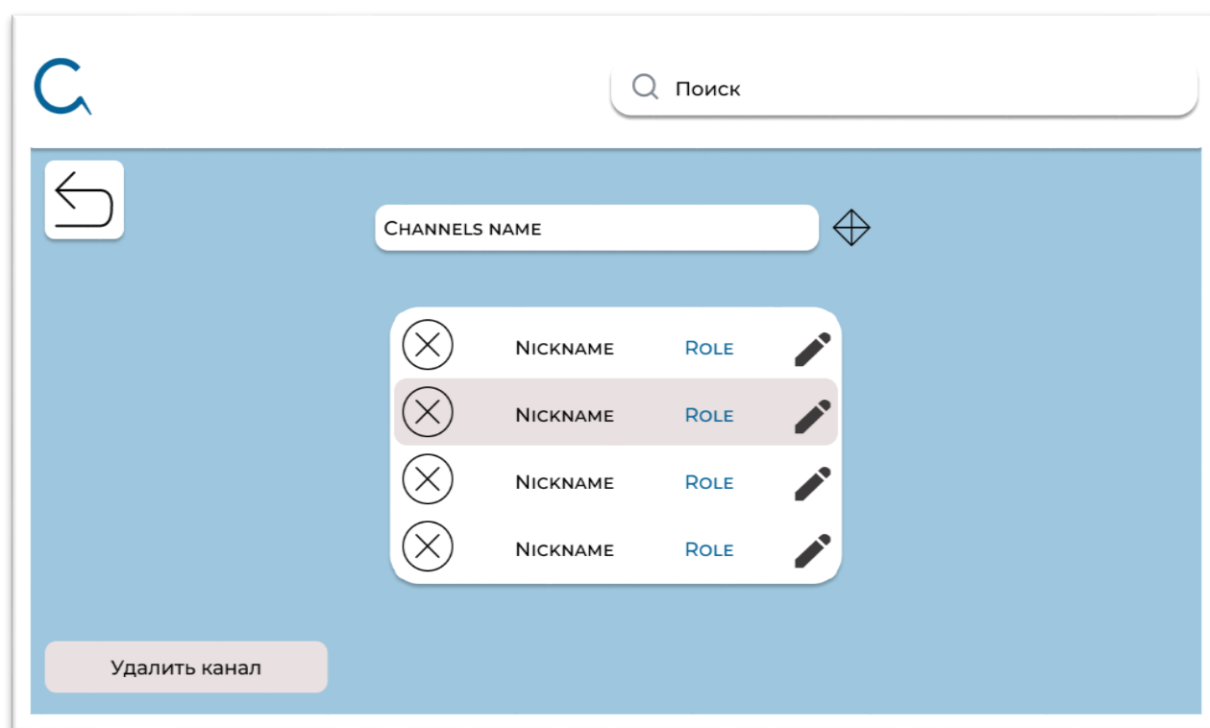


Рисунок 11 – Страница настроек канала для пользователя с правами администратора и создателя

2.3.7 Всплывающее окно добавления ролей

Пользователь (авторизованный, с правами администратора и создателя) имеет возможность добавить участнику канала роль.

Создатель имеет право изменять роль любого пользователя. Администратор не может изменить роль другому администратору.

Снизу расположен макет всплывающего окна с кнопками. При нажатии на соответствующую кнопку пользователь может: добавить права администратора, назначить роль пользователю.

The image shows a light gray rounded rectangle representing a modal window. Inside, at the top, is the text "Название роли" in a dark gray sans-serif font. Below this text is a white rounded rectangular input field. To the right of the input field is a small, empty black circle. At the bottom of the modal is a blue rounded rectangular button with the white text "Назначить".

Рисунок 12 – Всплывающее окно добавления ролей

2.3.8 Всплывающее окно создания канала

Пользователь (авторизованный) имеет возможность создать канал.

Снизу расположен макет всплывающего окна. При нажатии на соответствующую кнопку пользователь может создать канал.

The image shows a light gray rounded rectangle representing a modal window. Inside, at the top, is the text "Название канала" in a dark gray sans-serif font. Below this text is a white rounded rectangular input field. At the bottom of the modal is a blue rounded rectangular button with the white text "Создать".

Рисунок 13 – Всплывающее окно создания канала

2.3.9 Страница аккаунта пользователя

Пользователь (авторизованный) имеет возможность перейти на страницу своего аккаунта. Здесь у него есть возможность изменить аватар аккаунта, никнейм, пароль и почту.

Снизу расположен макет страницы. При нажатии на соответствующую кнопку пользователь может: изменить никнейм, изменить аватар, перейти на страницу изменения пароля или почты, вернуться на главную страницу.

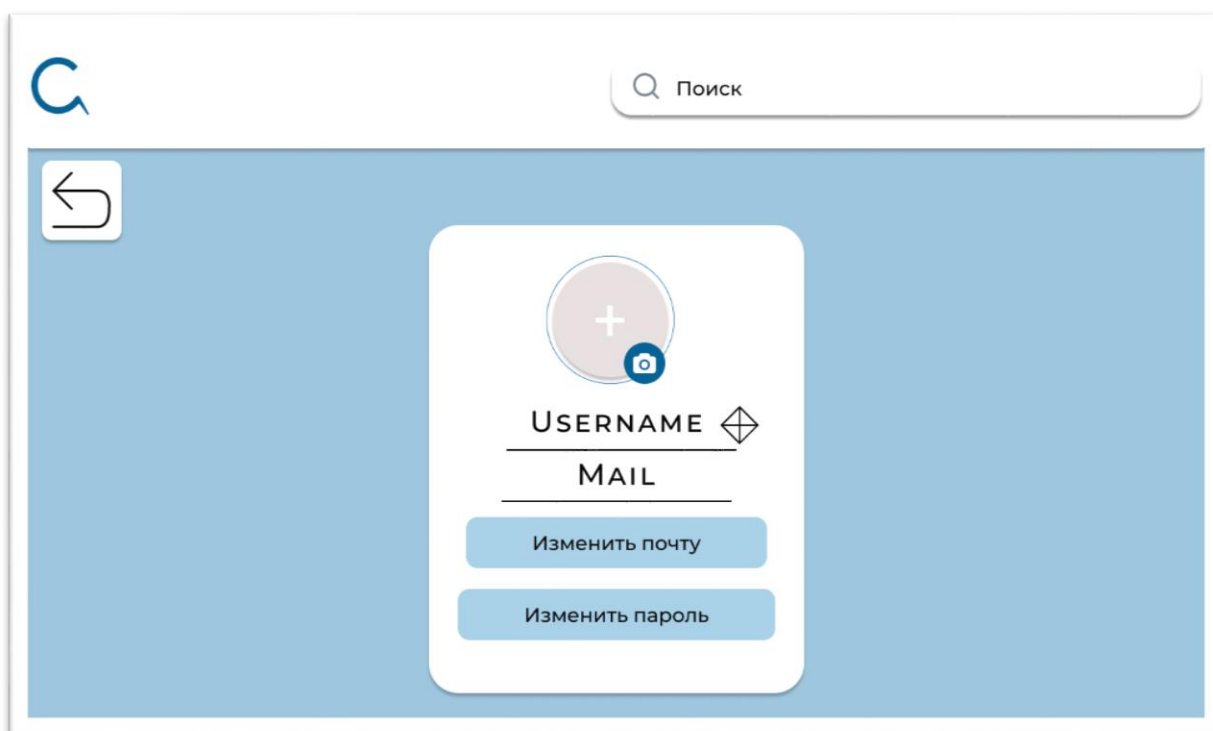


Рисунок 14 – Страница аккаунта пользователя

2.3.10 Страница изменения пароля

Пользователь (авторизованный) имеет возможность перейти со страницы своего аккаунта, на страницу изменения пароля. Здесь у него есть возможность изменить свой пароль.

Снизу расположен макет страницы. На ней есть поля для ввода старого пароля, и поля для ввода и подтверждения нового пароля. Кнопки для

подтверждения изменений и возвращения на главную страницу или страницу аккаунта.

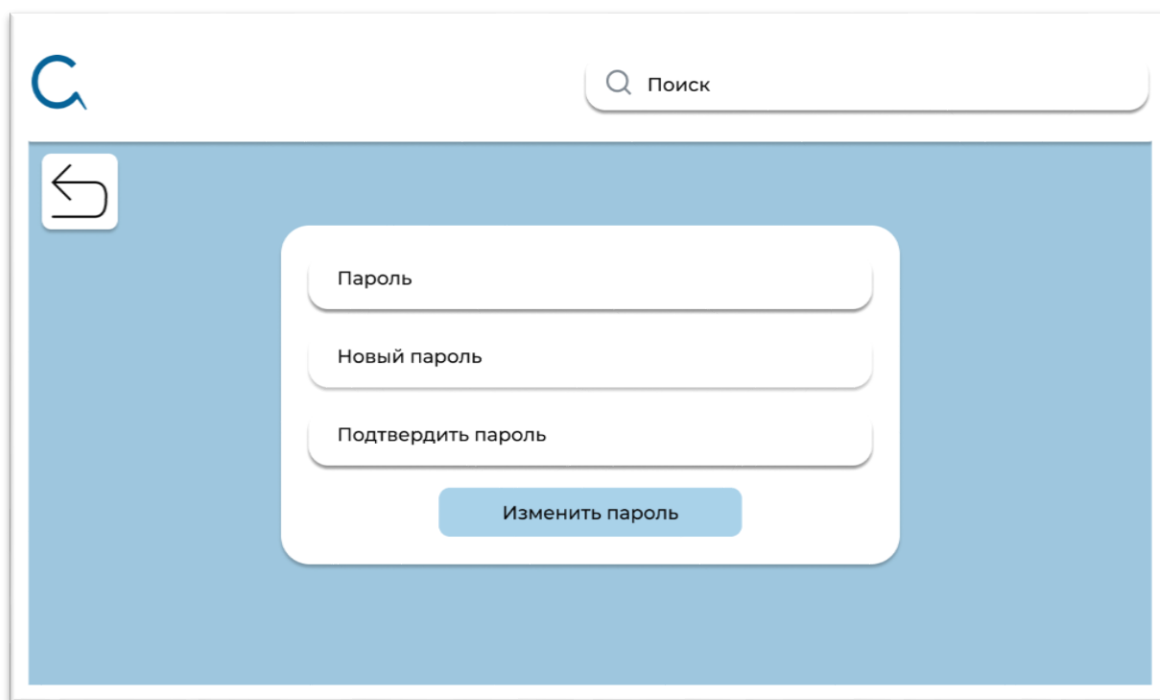


Рисунок 15 – Страница изменения пароля

2.3.11 Страница изменения почты

Пользователь (авторизованный) имеет возможность перейти со страницы своего аккаунта, на страницу изменения почты. Здесь у него есть возможность изменить свою почту.

Снизу расположен макет страницы. На ней есть поля для ввода новой почты, и поле для кода подтверждения, отправленного на старую почту. Кнопки для подтверждения изменений, отправки кода и возвращения на главную страницу или страницу аккаунта.

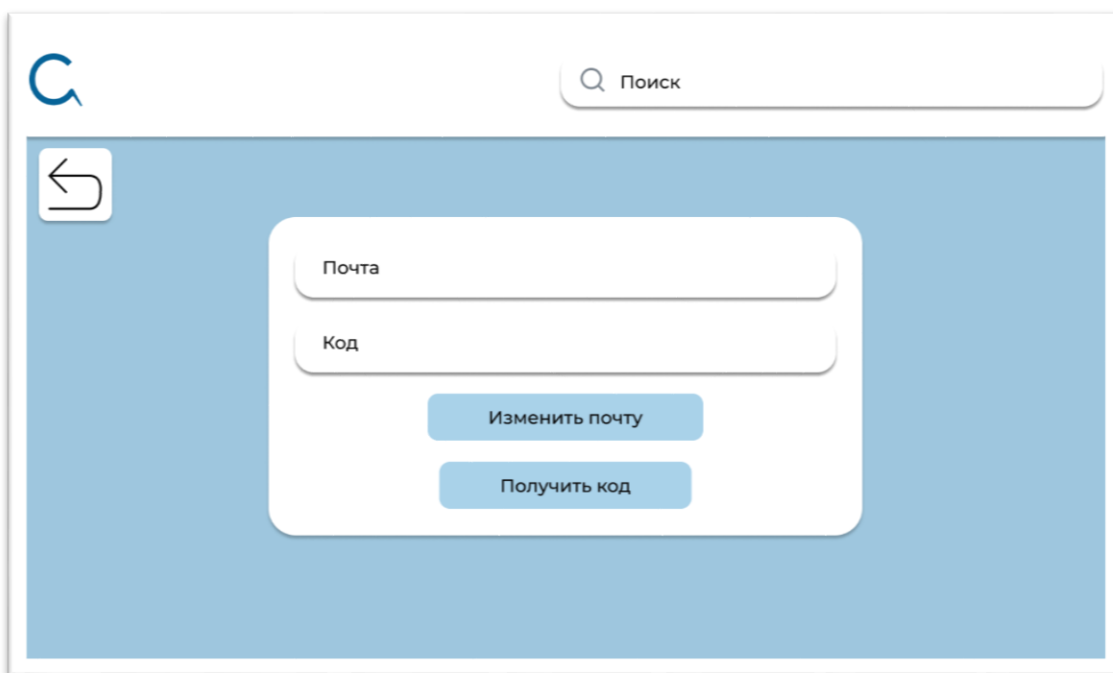


Рисунок 16 – Страница изменения почты

2.3.12 Страница входа в аккаунт

На данном экране отображены поля с вводом почты и пароля к аккаунту, кнопка «Войти», кнопка «Зарегистрироваться», кнопка для восстановления пароля, а также кнопка для возвращения на главную страницу для неавторизованного пользователя.

Снизу расположен макет страницы с кнопками. При нажатии на соответствующую кнопку пользователь переходит на страницы: регистрация, главная страница неавторизованного пользователя, главная страница авторизованного пользователя, восстановление пароля.

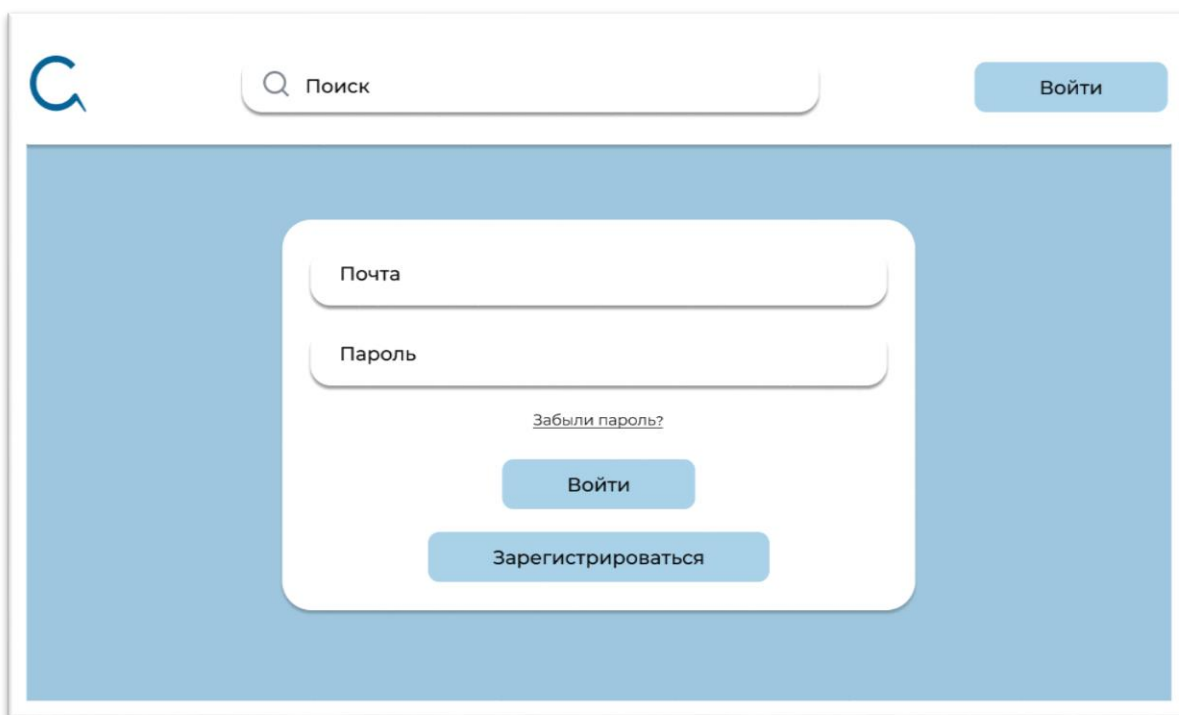


Рисунок 17 – Страница входа в аккаунт

2.3.13 Страница регистрации

На данном экране отображены поля с вводом никнейма, почты и пароля для нового аккаунта, кнопка «Зарегистрироваться», кнопка «Войти», а также кнопка для возвращения на главную страницу для неавторизованного пользователя.

Снизу расположен макет страницы с кнопками. При нажатии на соответствующую кнопку пользователь переходит на страницы: вход в аккаунт, главная страница неавторизованного пользователя.

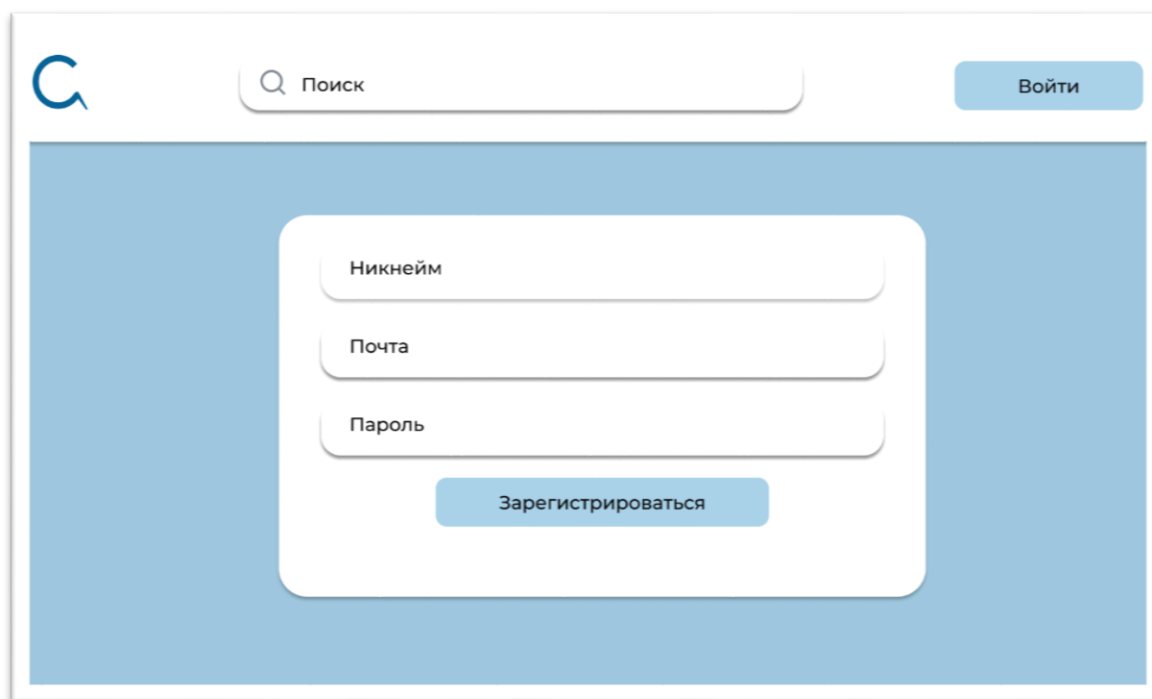


Рисунок 18 – Страница регистрации

2.3.14 Страница восстановление пароля

На данном экране отображены поля с вводом почты, нового пароля, подтверждением пароля и кодом подтверждения с почты пользователя, кнопка «Получить код», кнопка «Изменить пароль», а также кнопка для возвращения на главную страницу для неавторизованного пользователя.

Снизу расположен макет страницы с кнопками. При нажатии на соответствующую кнопку пользователь переходит на страницы: вход в аккаунт, главная страница неавторизованного пользователя. При нажатии кнопки «Получить код» пользователь получает код подтверждения для смены пароля на свою почту.

С

Поиск

Войти

Почта

Новый пароль

Подтвердить пароль

Код

Изменить пароль

Получить код

Рисунок 19 – Страница восстановление пароля

3 Диаграммы

3.1 Диаграмма прецедентов

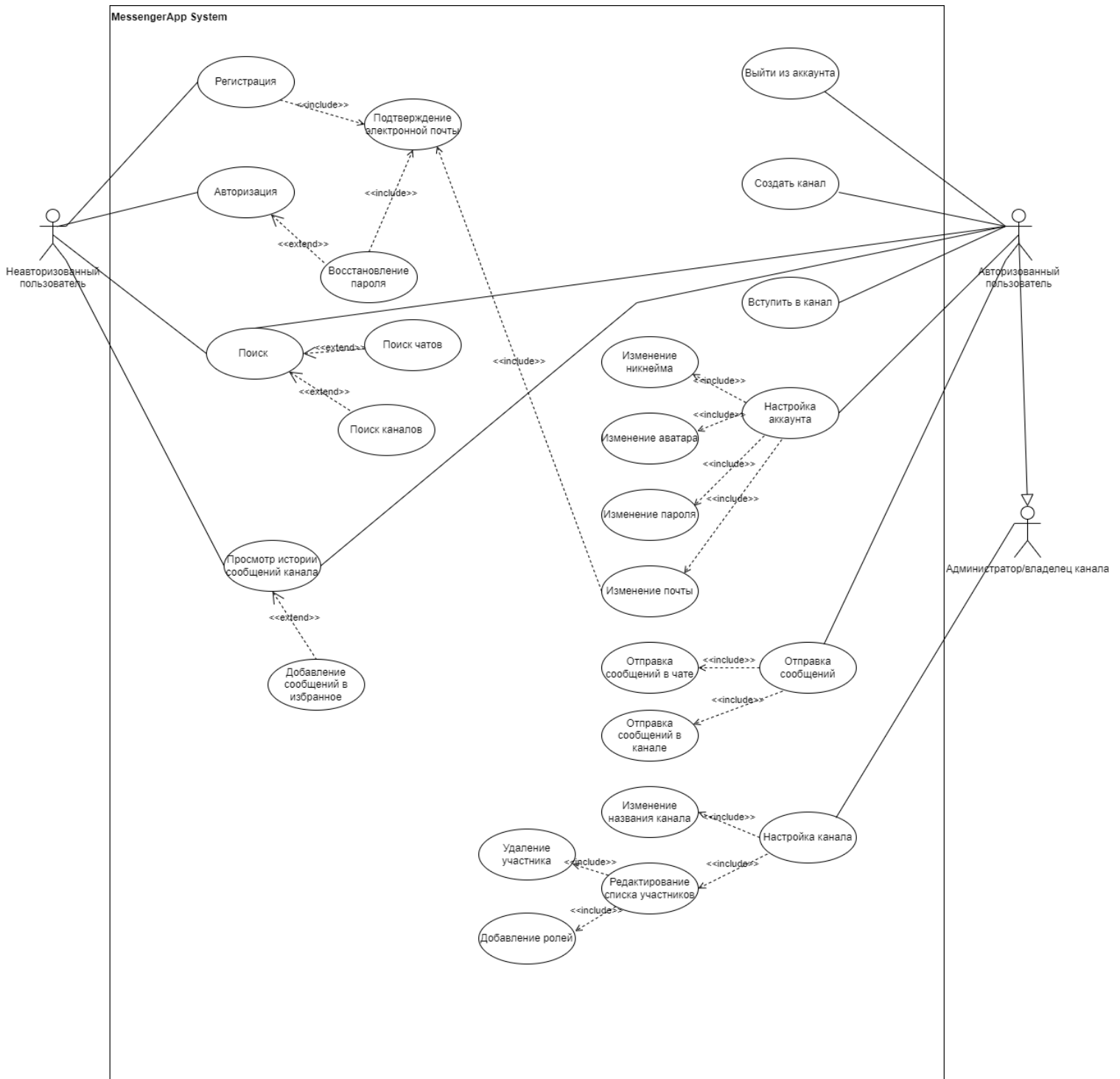


Рисунок 20 – Диаграмма прецедентов

На данной диаграмме представлены пользователи системы и доступные им действия. [2]

3.2 Диаграмма классов

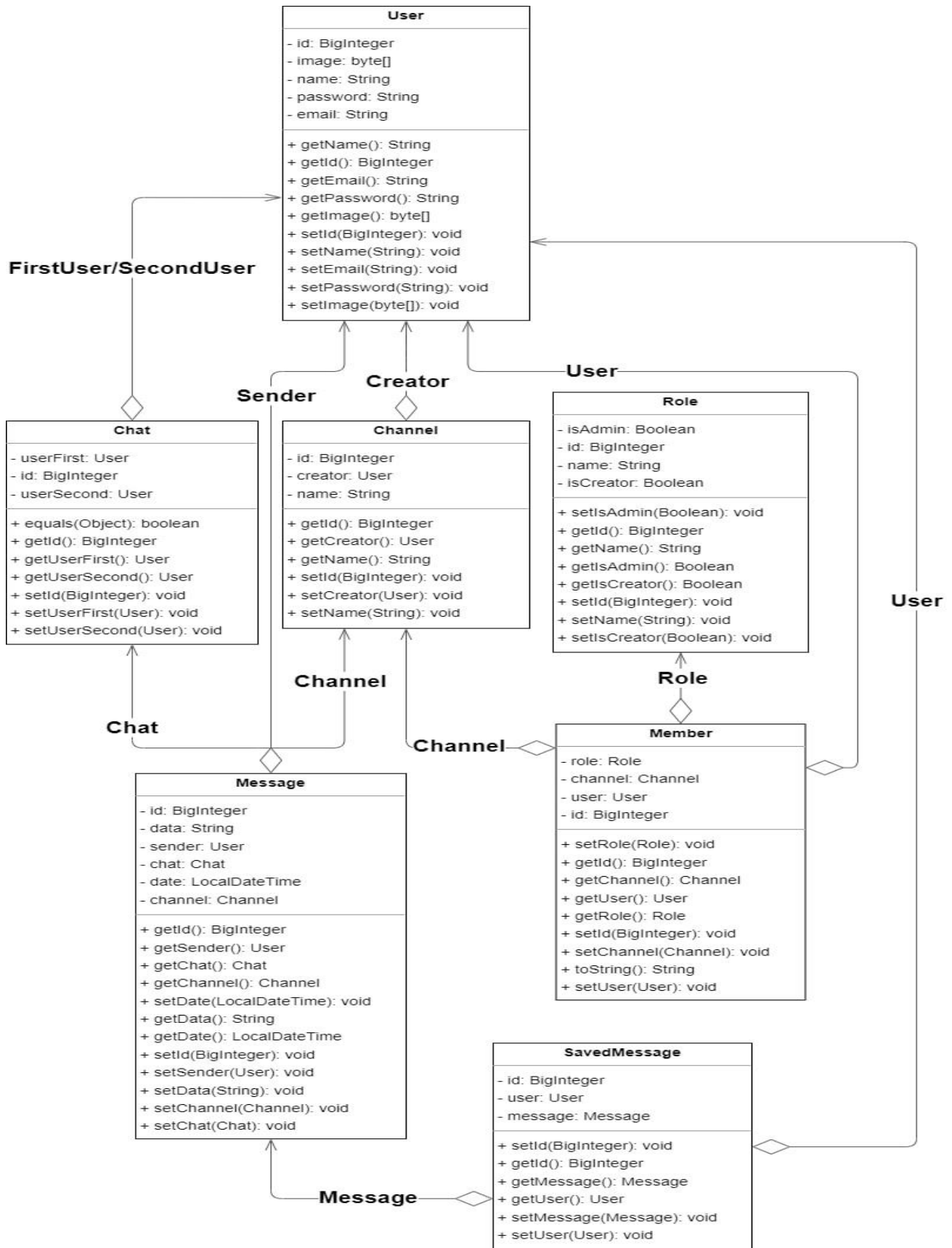


Рисунок 21 – Диаграмма классов

На данной диаграмме представлены классы системы, их методы и атрибуты с типами данных, также показано взаимодействие между классами посредством связей. [2]

3.3 Диаграмма последовательности

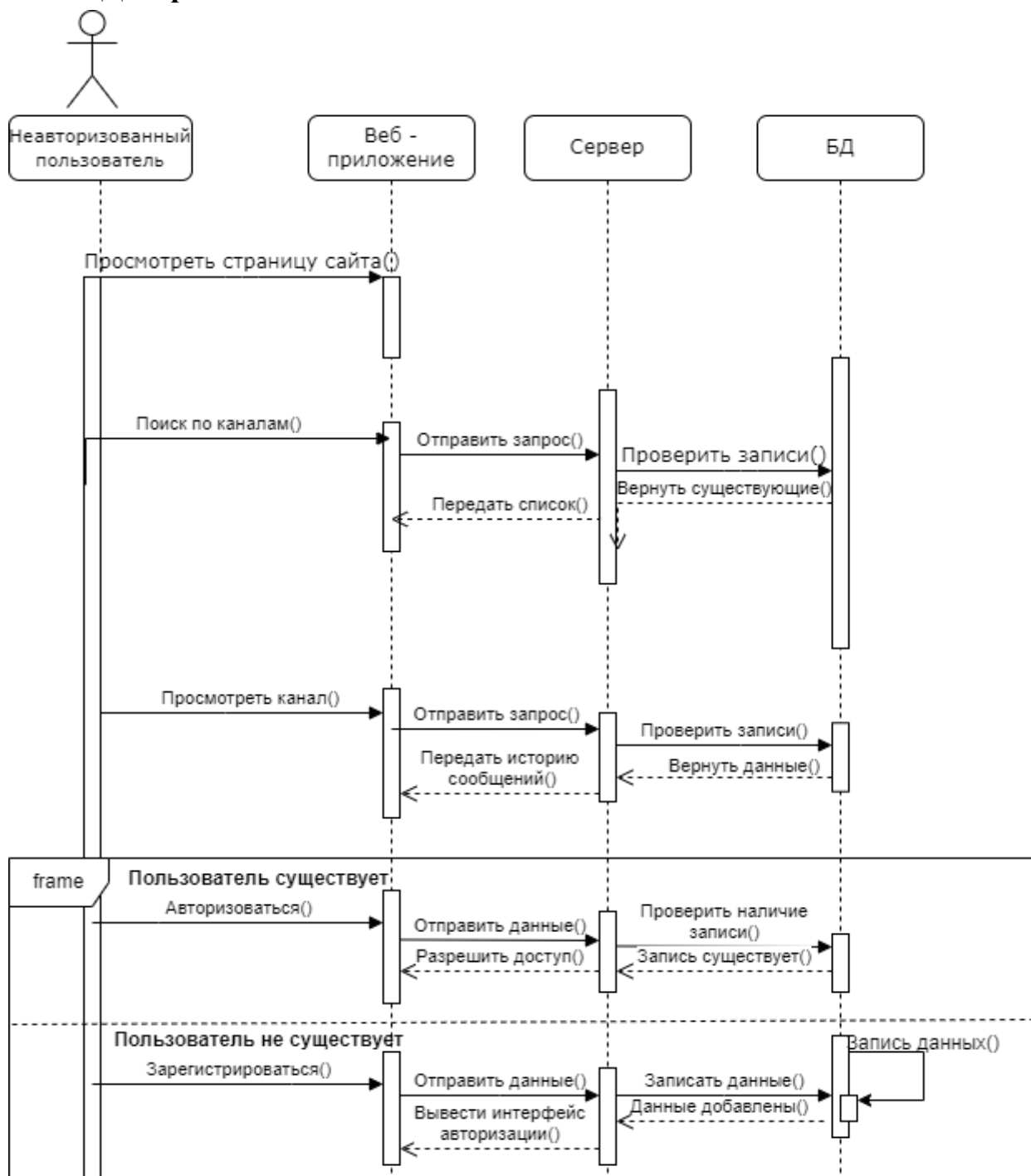


Рисунок 22 – Диаграмма последовательности для неавторизованного пользователя

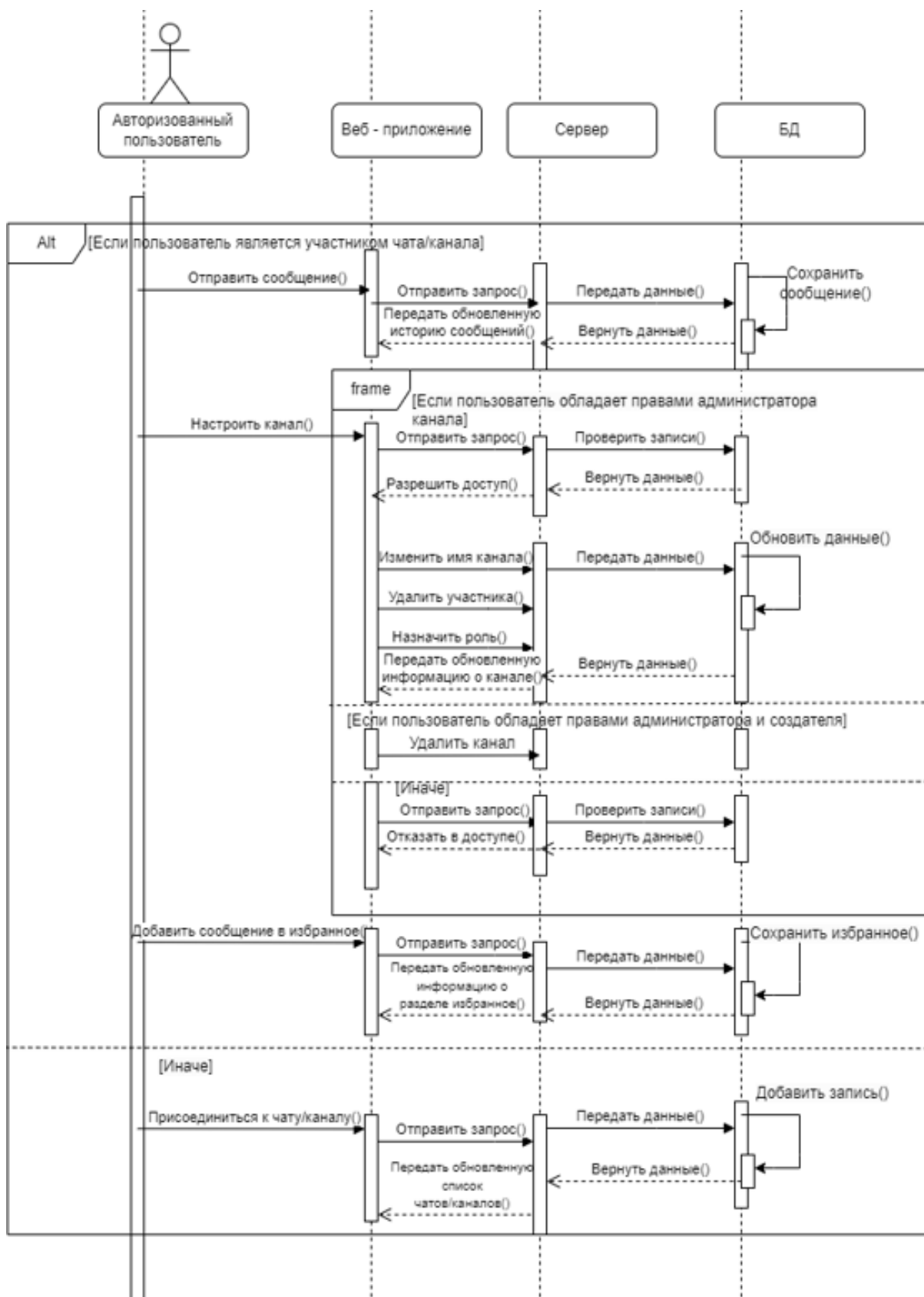


Рисунок 23 – Диаграмма последовательности со сценариями каналов для авторизованного пользователя

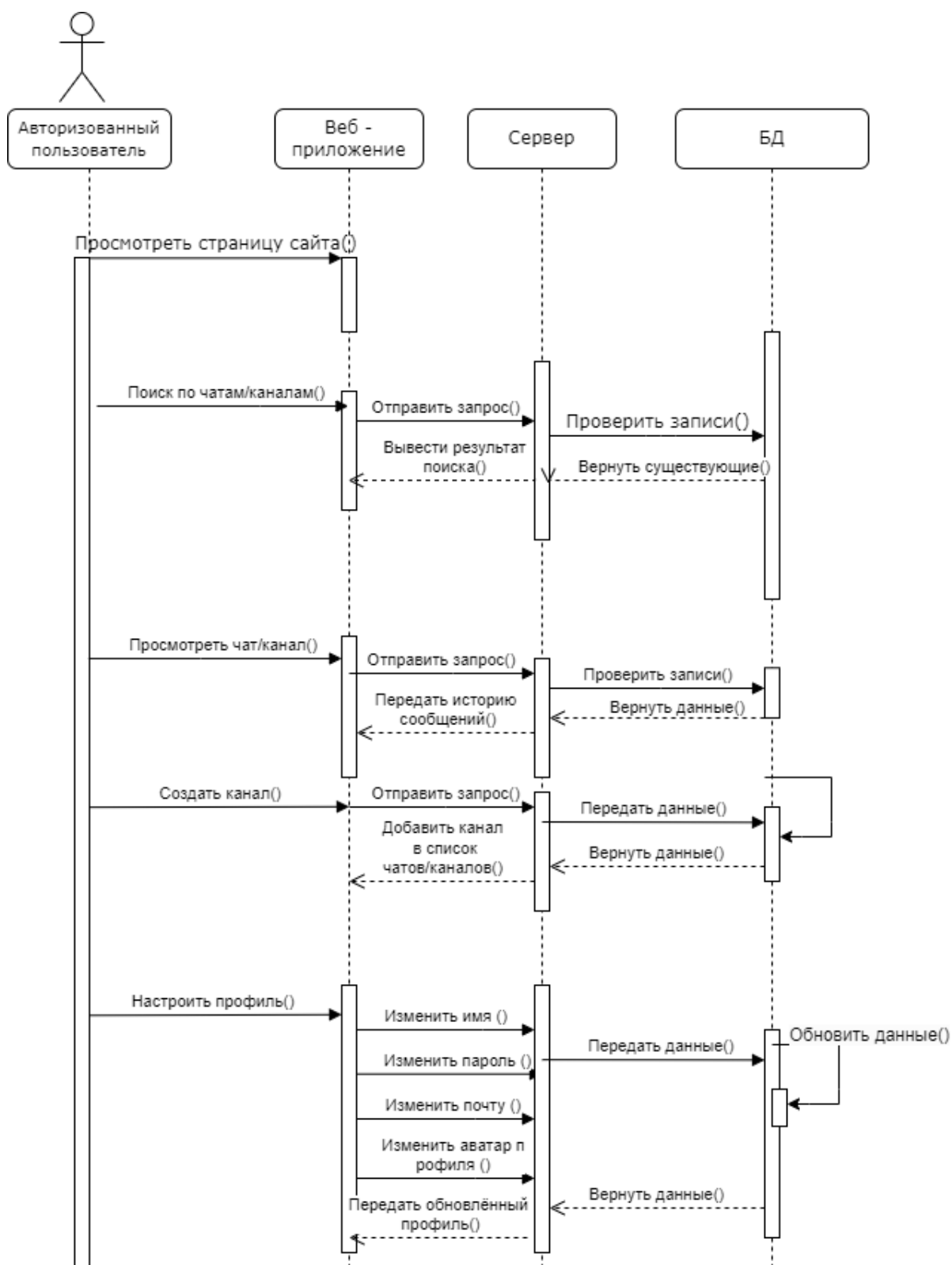


Рисунок 24 – Диаграмма последовательности без сценариев каналов для авторизованного пользователя

На данной диаграмме представлены взаимодействия различных частей системы между собой для выполнения функции, данная диаграмма также показывает последовательность действий, которые приводят к завершению этих функций. [2]

3.4 Диаграмма активности

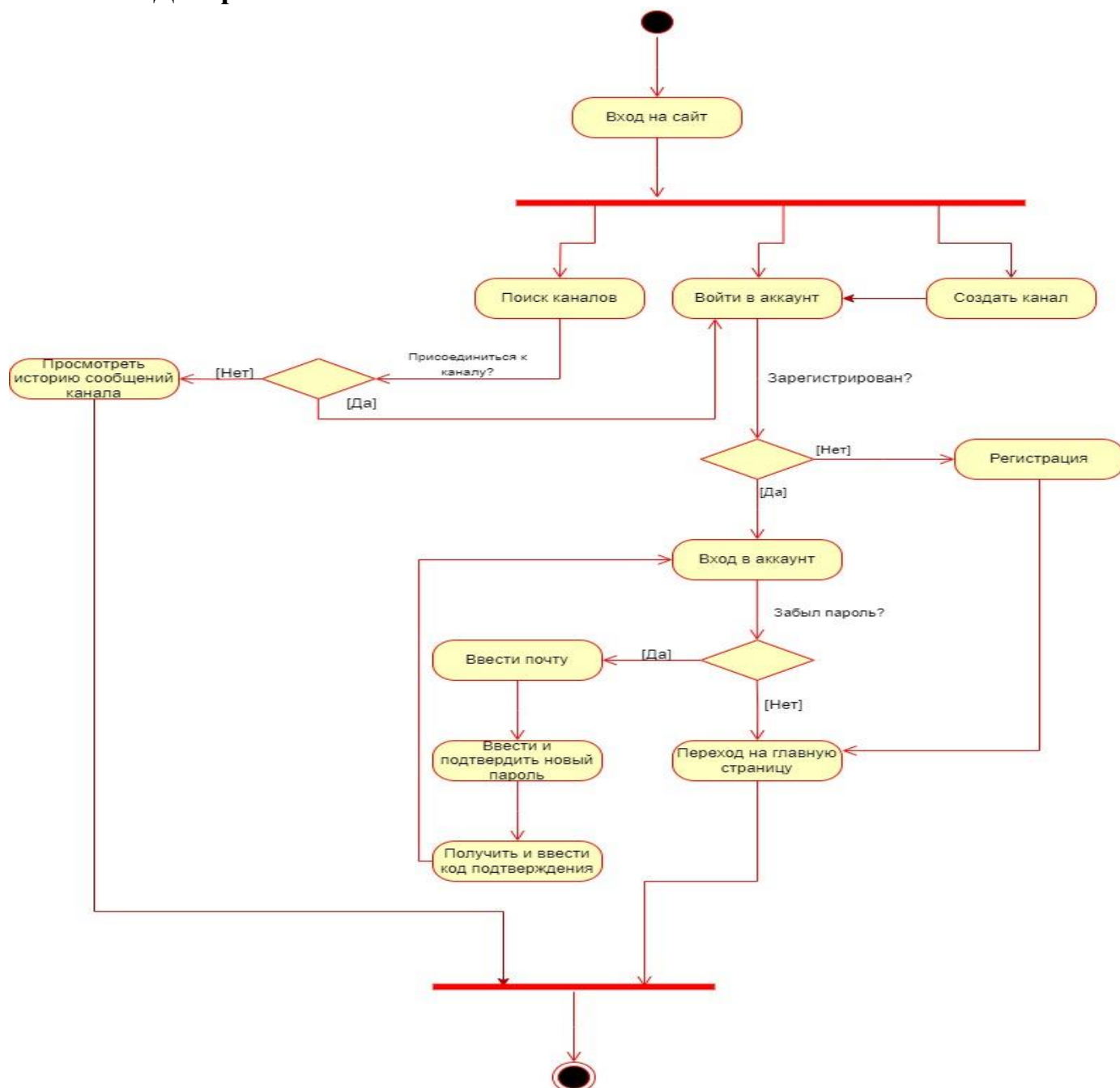


Рисунок 25 – Диаграмма активности для неавторизованного пользователя

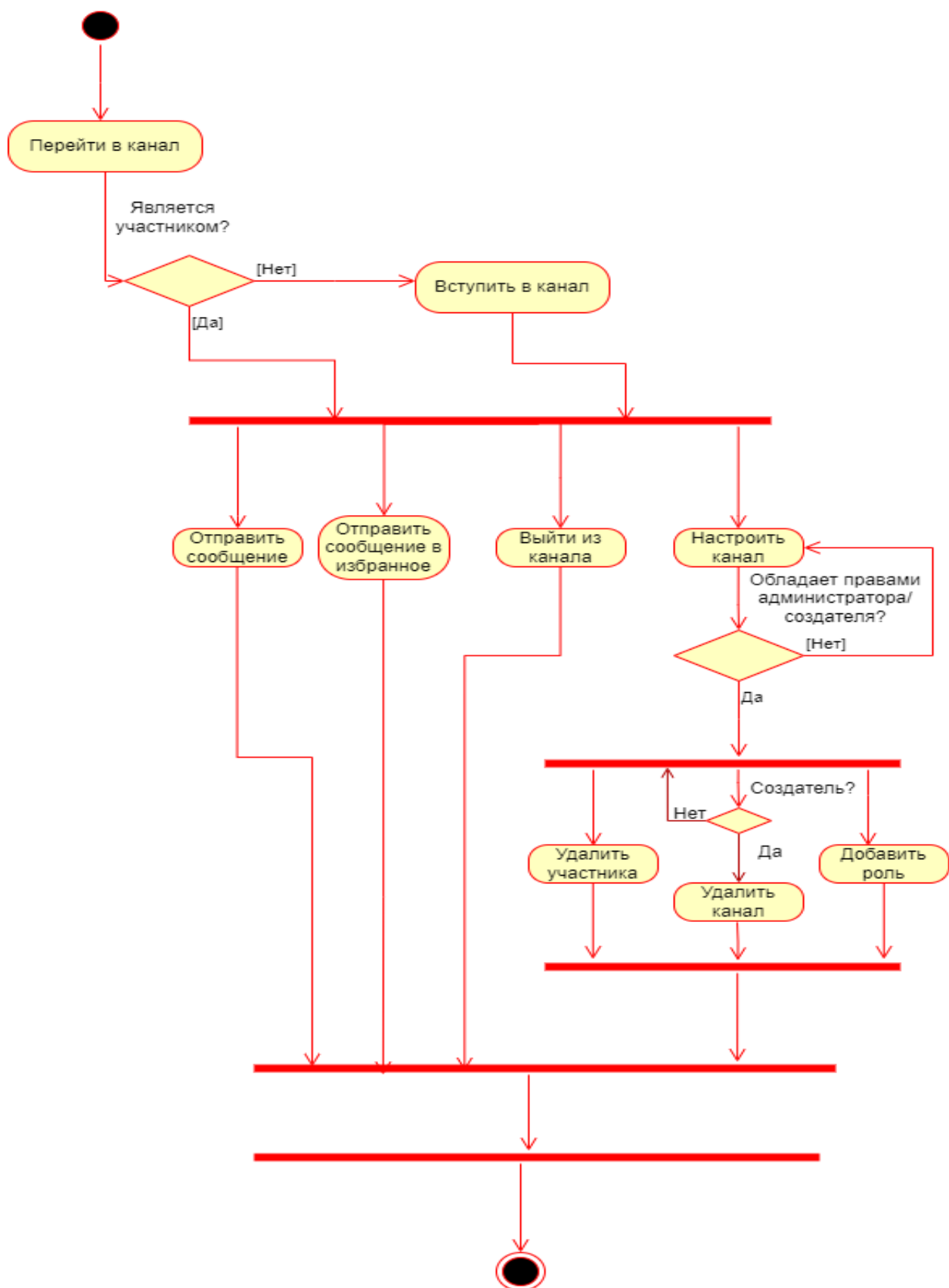


Рисунок 26 – Диаграмма активности со сценариями каналов для авторизованного пользователя

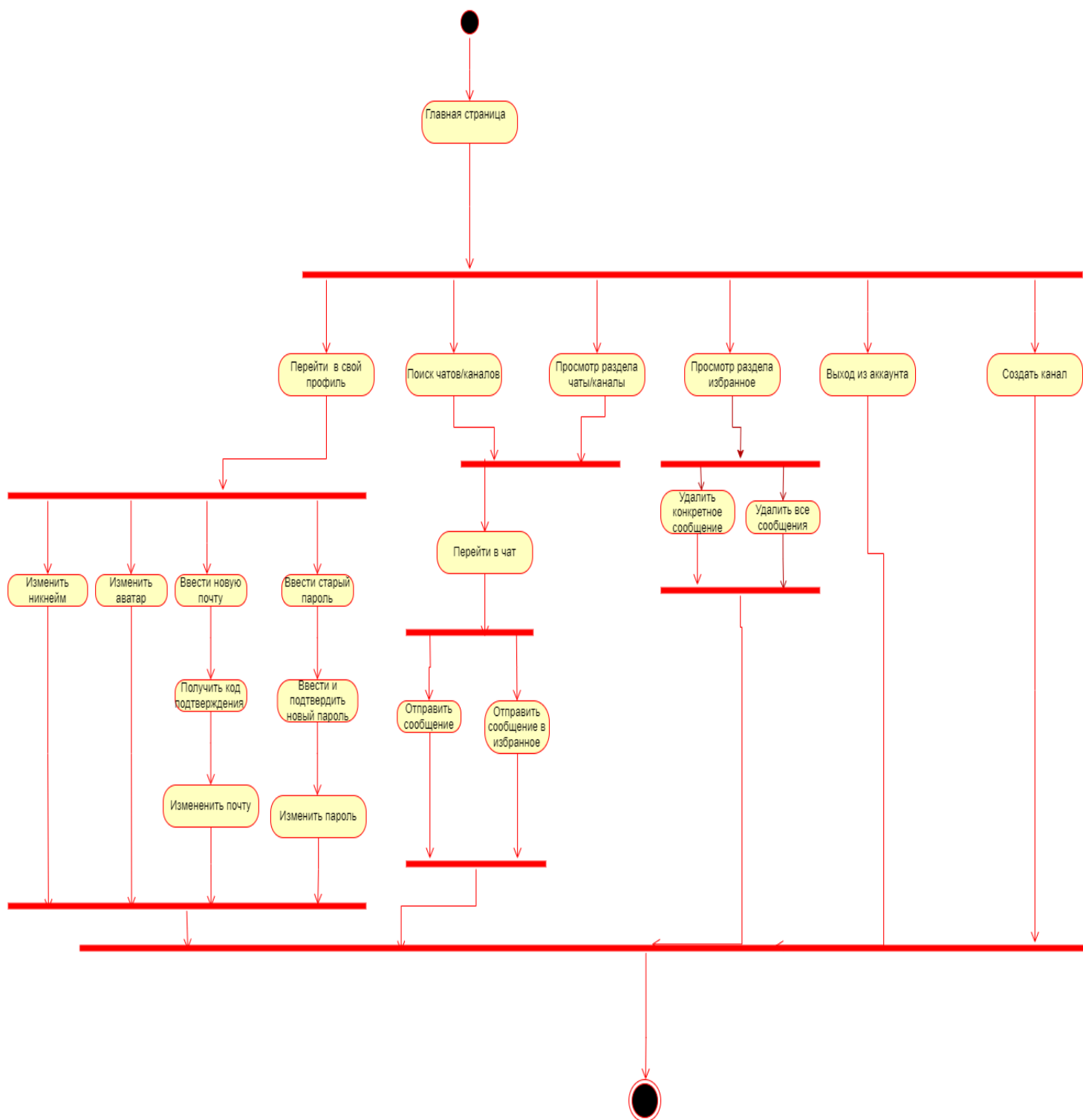


Рисунок 27 – Диаграмма активности без сценариев каналов для авторизованного пользователя

На данной диаграмме представлены возможные действия пользователей и их последовательность. [2]

3.5 Диаграмма развёртывания

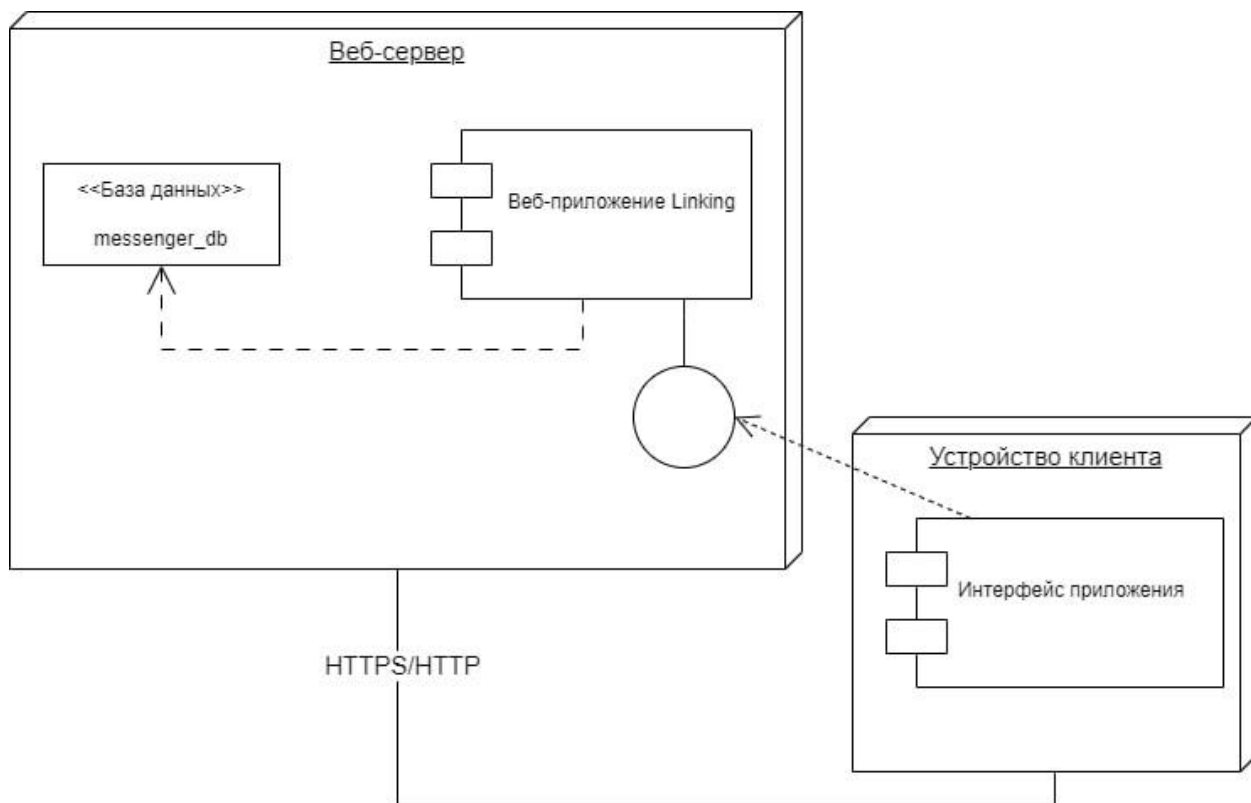


Рисунок 28 – Диаграмма развёртывания

На данной диаграмме представлены основные программные компоненты, способы их развёртывания и общая конфигурация проектируемой программной системы. [2]

3.6 Диаграмма сотрудничества

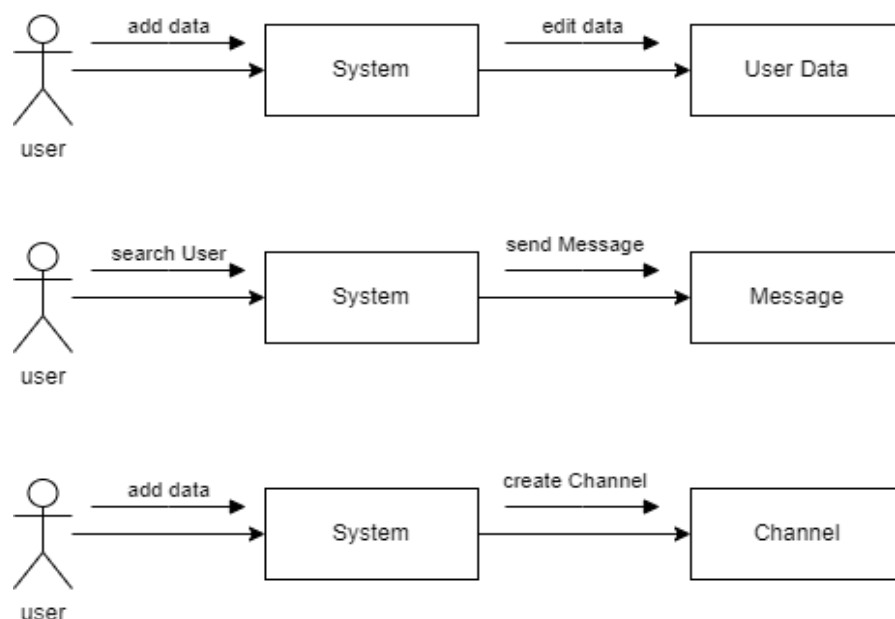


Рисунок 26 – Диаграмма сотрудничества

На данной диаграмме представлено соединение и взаимодействие объектов между собой в системе. [2]

3.7 Диаграмма объектов

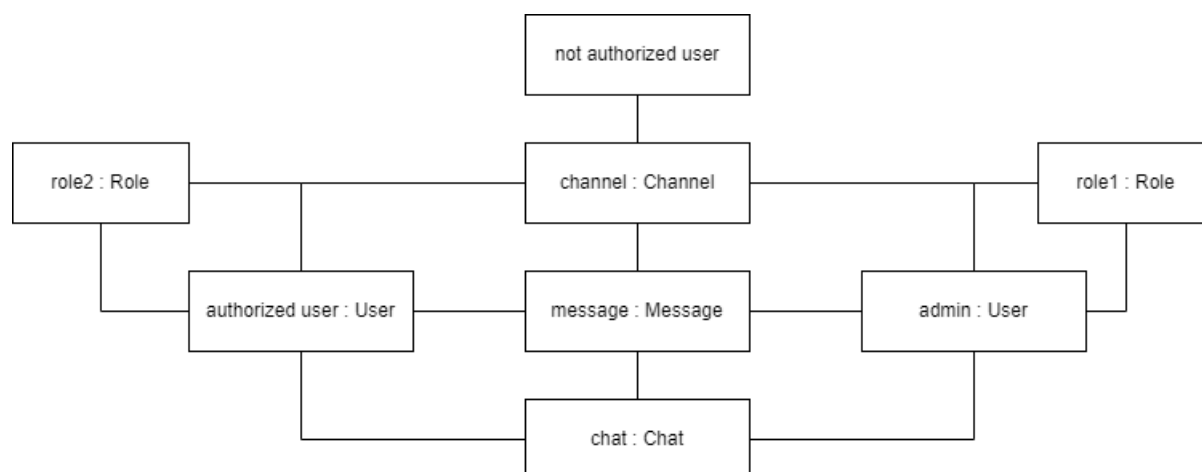


Рисунок 27 – Диаграмма объектов

На данной диаграмме представлены объекты, используемые данной системой. [2]

3.8 Диаграмма состояний

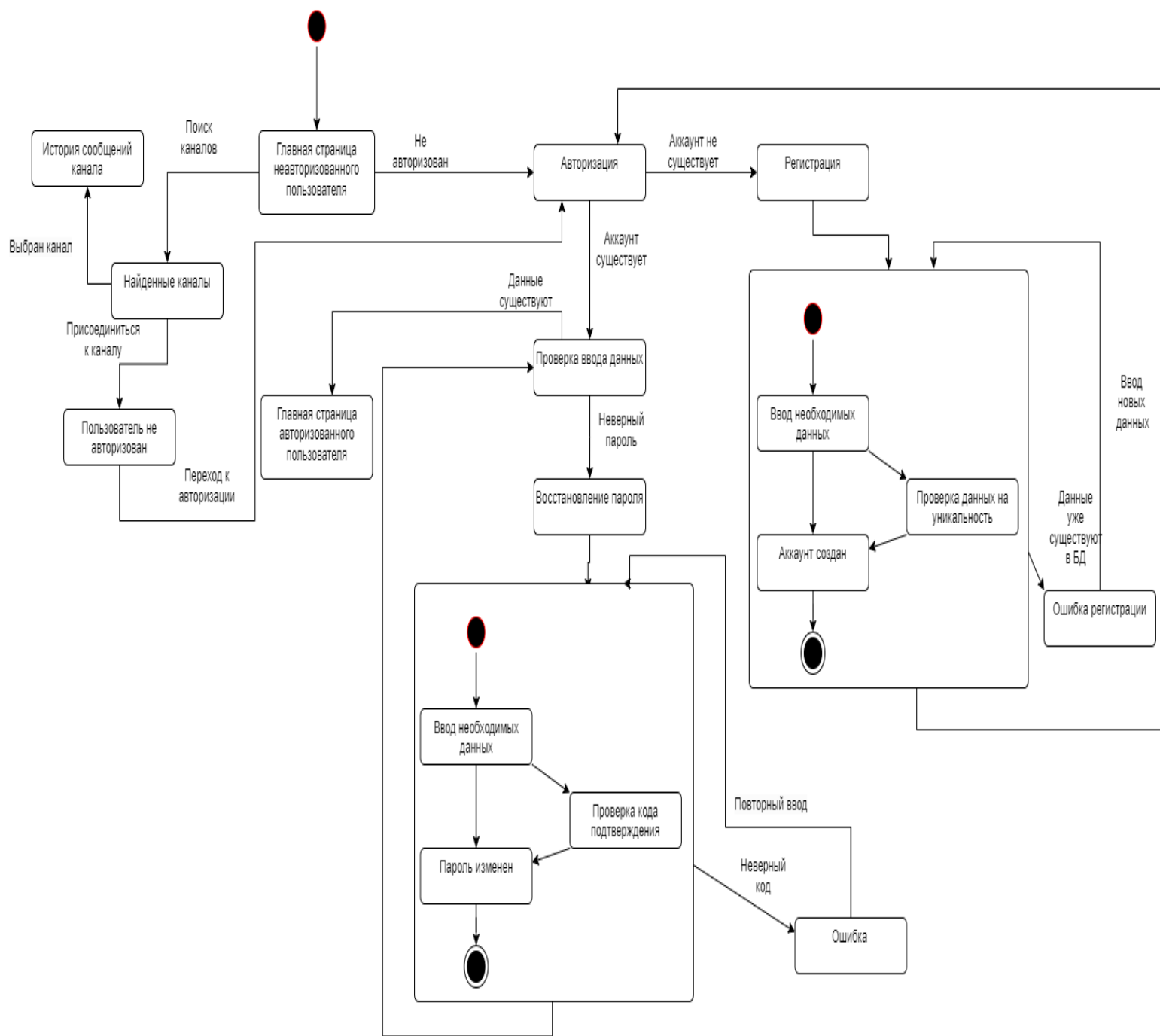


Рисунок 28 – Диаграмма состояния для неавторизованного пользователя

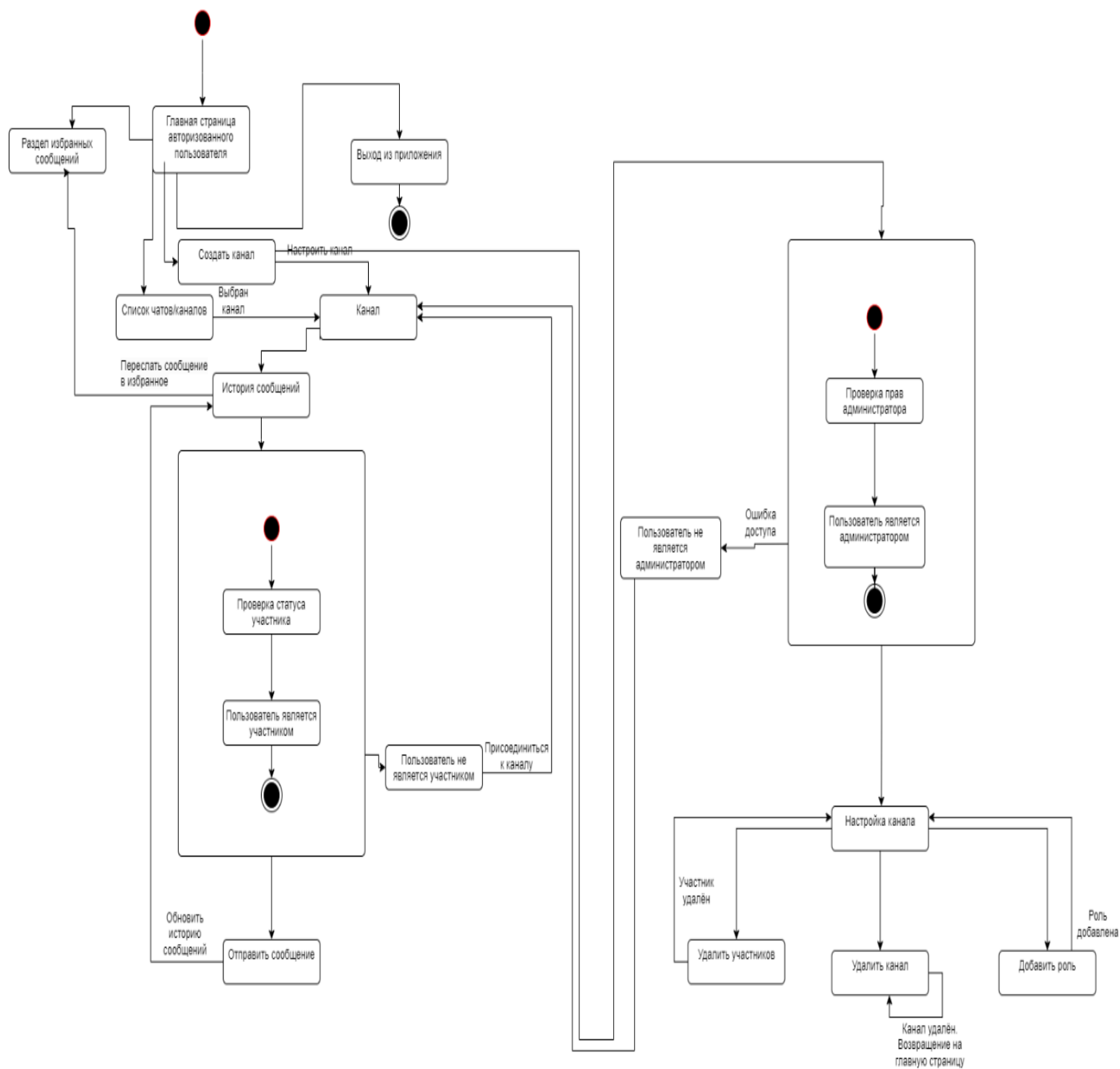


Рисунок 29 – Диаграмма состояния со сценариями каналов для авторизованного пользователя

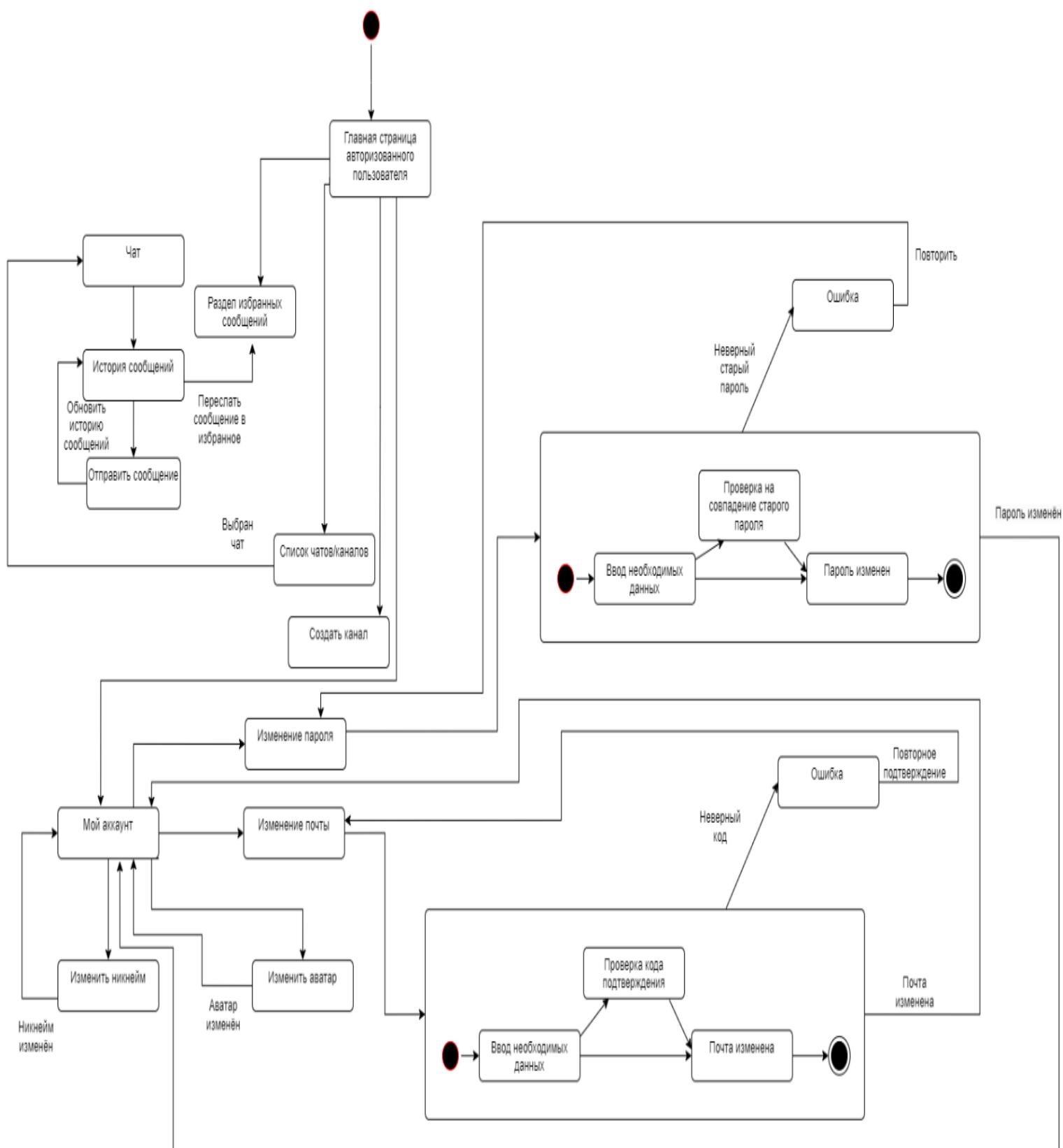


Рисунок 30 – Диаграмма состояния без сценариев каналов для авторизованного пользователя

На данной диаграмме представлены разрешенные состояния и переходы, а также события, которые влияют на эти переходы. [2]

3.9 ER-Диаграмма

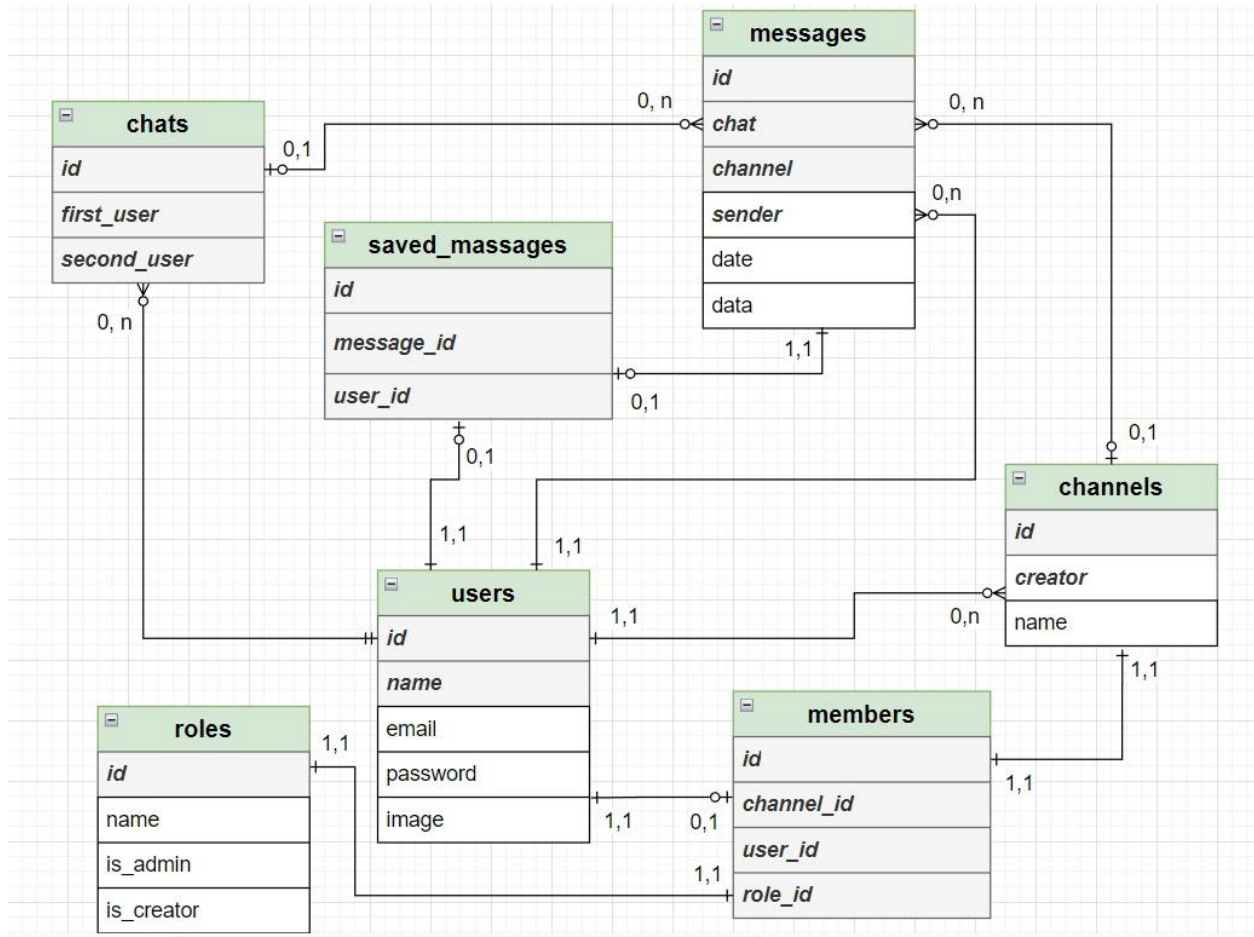


Рисунок 31 – ER-Диаграмма

На данной диаграмме представлена база данных, которая показывает, как связаны элементы внутри. В данном случае она иллюстрирует, какие есть сущности и как они связаны внутри системы разрабатываемого веб-приложения. [2]

3.10 Физическая схема базы данных

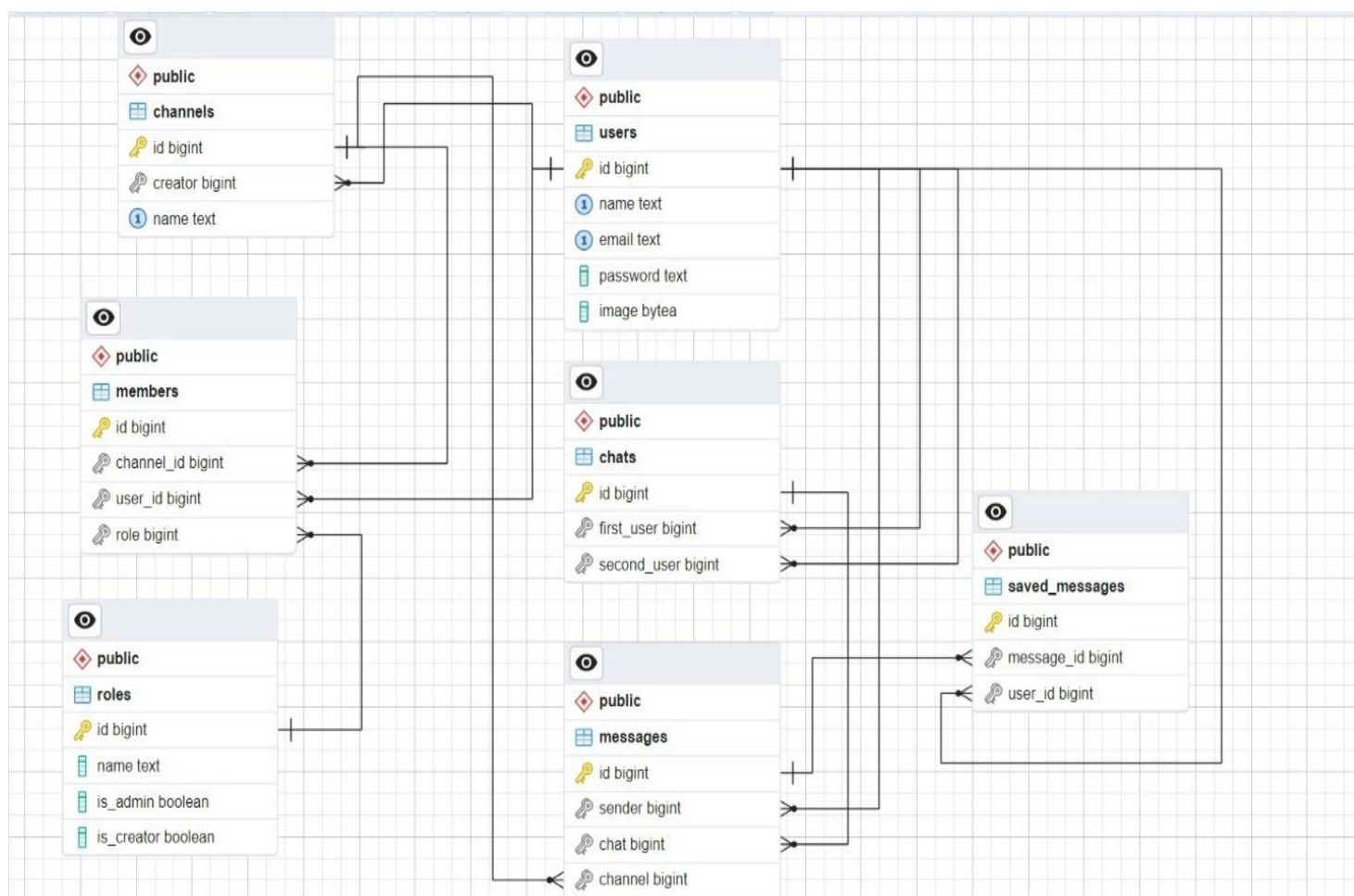


Рисунок 32 – Физическая схема базы данных

На данной диаграмме представлена модель данных, которая определяет, каким образом представляются данные, и содержит все детали, необходимые СУБД для создания базы данных. [2]

3.11 IDEF0 Диаграмма

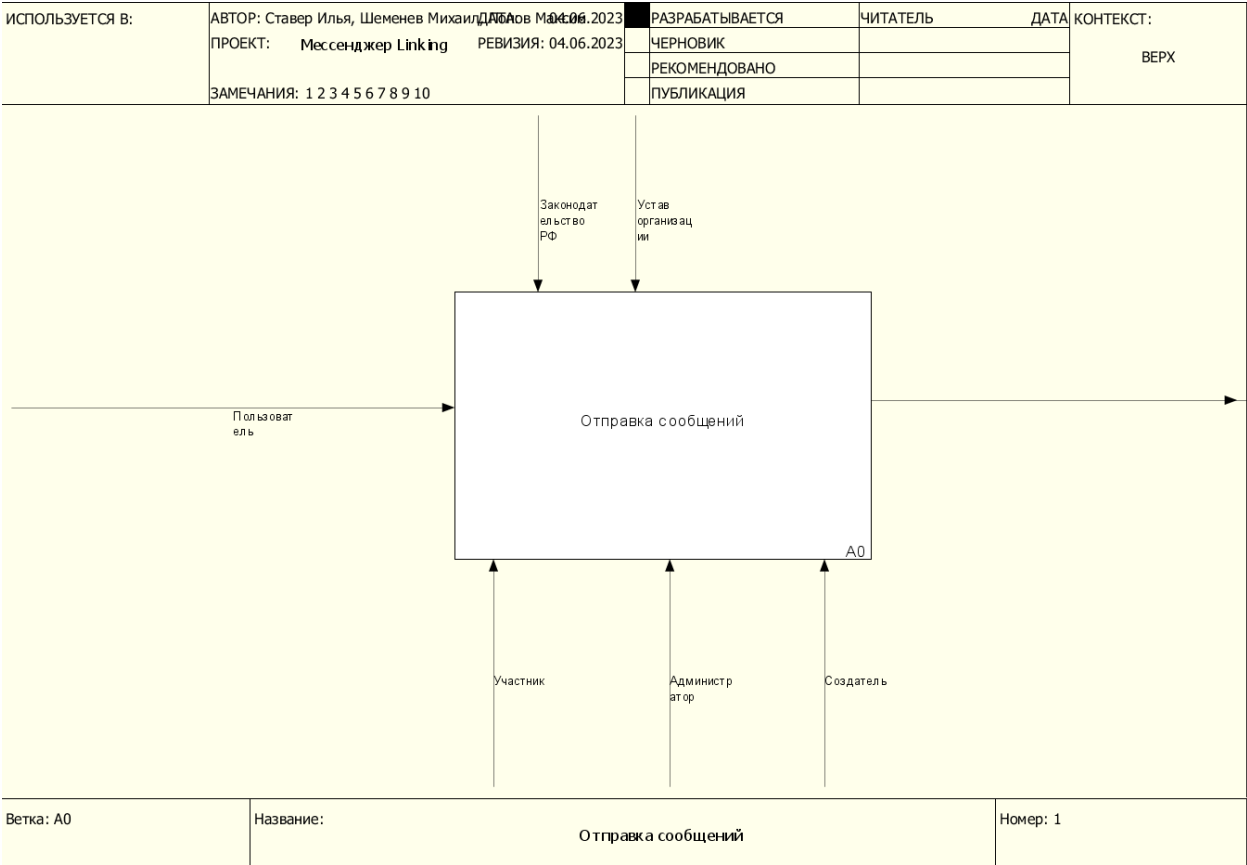


Рисунок 33 – IDEF0 Диаграмма (0 уровень декомпозиции)

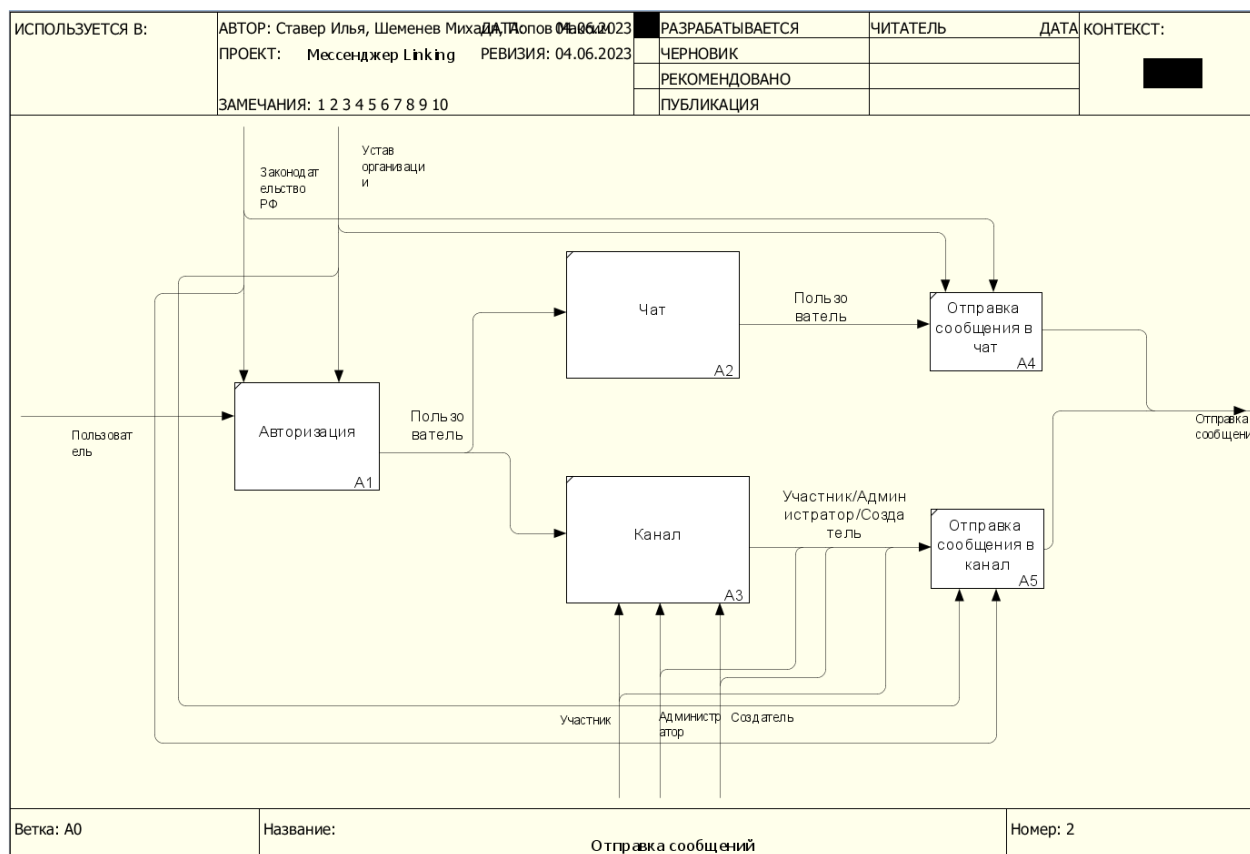


Рисунок 34 – Диаграмма IDEF0(1 уровень декомпозиции)

На данной диаграмме представлены структура и функции системы, а также потоки информации и материальных объектов, связывающих эти функции. [2]

4 Реализация

4.1 Средства реализации

Для разработки мессенджера был выбран следующий стек технологий:

- Java – строго типизированный объектно-ориентированный язык программирования. Был выбран в качестве основного, так как он остается очень популярным языком программирования в этой области благодаря своим мощным возможностям и широкому спектру инструментов для разработки. К тому же существует огромное количество фреймворков и библиотек, написанных на Java, которые в перспективе можно легко интегрировать в проект;
- Javascript – мультипарадигменный язык программирования. Данный язык был выбран в качестве основного, так как его характеризует простота и рациональность применения, комфортность использования пользовательских интерфейсов, а также наличием собственной мощной инфраструктуры. Кроме того, он является кроссплатформенным, что позволяет один и тот же код адаптировать как для компьютеров, так и для мобильных устройств;
- Swagger – инструмент для документирования и тестирования API. Он позволяет создавать интерактивную документацию для вебсервисов, что упрощает их использование и интеграцию. Swagger автоматически генерирует документацию на основе аннотаций в коде, что позволяет разработчикам сосредоточиться на написании логики приложения, а не на создании и поддержке документации. Благодаря Swagger, разработчики могут изучить доступные эндпоинты, параметры, модели данных и примеры запросов и ответов; [3]

- Render – это удобный для разработчиков хост, который позволяет разворачивать практически все в облаке, включая статические сайты, веб-приложения, файлы Dockerfile, API и базы данных PostgreSQL; [4]
- FlyWay – продукт с открытым исходным кодом для обеспечения миграций баз данных. Был выбран, т.к. легко интегрируется со Spring Framework и поддерживает PostgreSQL 14;
- PostgreSQL – объектно-реляционная система управления базами данных, основанная на языке SQL; [5]
- Spring Boot Framework – универсальный фреймворк с открытым исходным кодом для Java-платформы. Был выбран, так как он совместим с большим количеством библиотек и фреймворков, что позволяет использовать его в различных проектах и на различных платформах. Так же он позволяет разработчикам быстро создавать приложения без необходимости тратить много времени на конфигурацию; [6]
- Docker – это программная платформа для быстрой разработки, тестирования и разворачивания приложений;
- React-Native Framework – кроссплатформенный фреймворк для разработки нативных мобильных и настольных приложений; [7]
- CSS – язык стилей, используемый для оформления веб-страниц. Он определяет внешний вид и расположение элементов на странице, включая цвета, шрифты, размеры, отступы, границы и другие стилевые свойства. Он обладает следующими преимуществами: возможность изменения внешнего вида всего сайта путем изменения одного файла CSS, возможность переиспользования стилей благодаря созданию классов и переопределения стилей для различных элементов;

- HTML – язык разметки, используемый для создания структуры и содержимого веб-страниц. Он обладает следующими преимуществами: читаемость для разработчиков, возможность создания структурированного контента с использованием семантических элементов, доступность и адаптивности для различных устройств и браузеров. HTML является основным языком для создания веб-страниц и работает в сочетании с CSS для определения внешнего вида и JavaScript для добавления интерактивности;
- Vercel – облачная платформа для развертывания веб-приложений. Она обладает следующими преимуществами: масштабируемость, автоматическое развертывание. [8]
- Miro – платформа для совместной работы распределенных команд;
- Draw.io – Бесплатное кроссплатформенное программное обеспечение для рисования графиков с открытым исходным кодом. Его интерфейс можно использовать для создания диаграмм, таких как блок-схемы, каркасы, диаграммы UML;
- BPwin – графическая среда для проектирования и моделирования сложных систем широкого назначения, который может быть использован для создания диаграмм в формате IDEF0;
- Figma – онлайн-сервис для дизайнеров, веб-разработчиков и маркетологов. Он предназначен для создания прототипов сайтов или приложений, иллюстраций и векторной графики.
- Git – распределённая система управления версиями;
- GitHub – платформа разработки программного обеспечения с открытым исходным кодом, представляющая систему управления репозиториями кода для Git;

— Trello – визуальный инструмент, обеспечивающий эффективность командной работы на любом проекте.

4.2 Реализация Backend

Серверная (backend) часть приложения была написана на языке Java с использованием фреймворка Spring Boot. Spring Boot подразумевает разделение приложения на модули (прослойки), которые будут описаны далее. Каждый модуль организован в виде отдельного пакета.

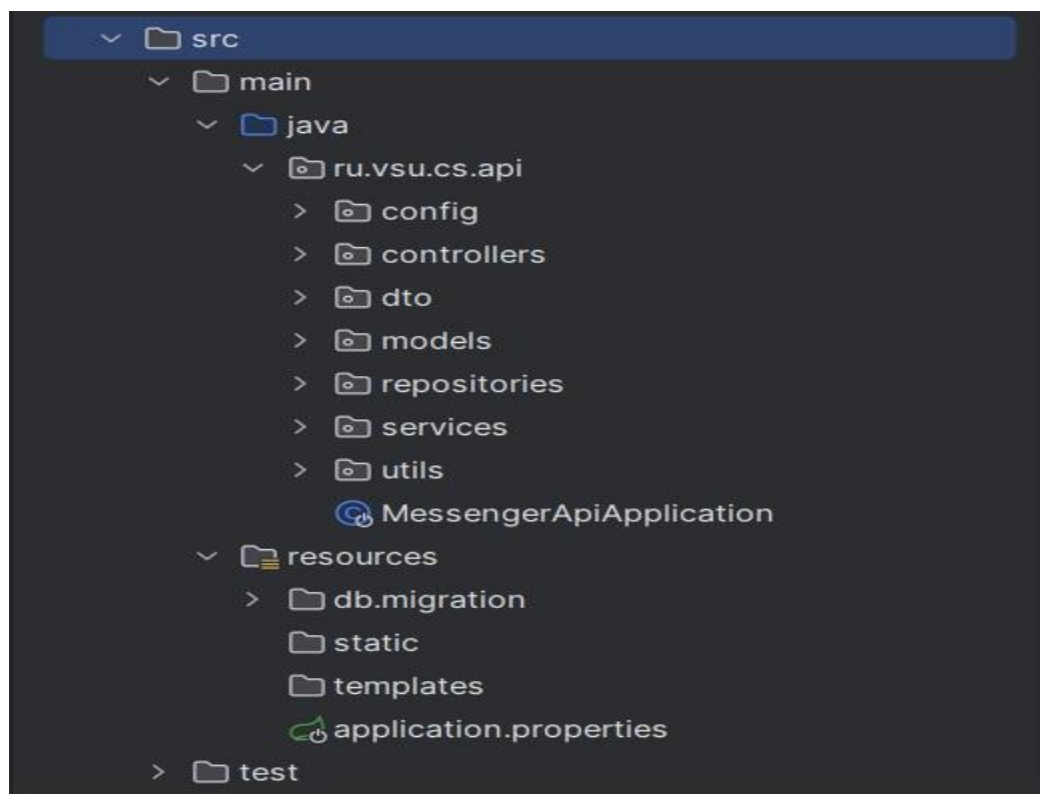


Рисунок 30 - Структура серверной части приложения

Приложение разделено на следующие модули:

— Модуль приложения (Application module). Это основной модуль, который содержит классы и настройки, связанные с запуском и конфигурацией приложения Spring Boot. В этом модуле находится главный класс приложения с аннотацией `@SpringBootApplication` и файл конфигурации

application.properties, содержащий параметры подключения к базе данных. Для защиты базы данных эти параметры не объявляются явно в коде, а передаются в программу через переменные окружения;

- Модуль контроллеров (Controller Module). В этом модуле находятся классы контроллеров, отвечающие за обработку запросов и взаимодействие с клиентом (браузером). Контроллеры содержат аннотации, такие как @RestController и @RequestMapping, для определения эндпоинтов и обработки запросов;
- Модуль конфигурации (Configuration Module). В этом модуле находятся классы конфигурации Spring, которые содержат настройки и бины, связанные с приложением. В случае разрабатываемого приложения в модуле конфигурации есть только один класс SecurityConfig, в котором определены и настроены основные компоненты Spring Security для обеспечения безопасности серверной части приложения. Данный файл конфигурации настраивает основные компоненты безопасности Spring Security, а именно шифрование паролей (используется BCryptPasswordEncoder) и правила авторизации (позволяют определить права доступа пользователя к различным ресурсам);
- Модуль моделей (Model Module). Этот модуль содержит классы моделей данных, которые отображают структуру таблиц в базе данных и представляют объекты бизнес-логики. Модели аннотированы аннотациями JPA, такими как @Entity и @Table, и содержат поля, геттеры и сеттеры. При работе с базой данных использовался фреймворк Hibernate. Он обеспечил автоматическое отображение классов и свойств на таблицы и столбцы в базе данных;

- Модуль репозитория (Repository Module). В этом модуле находятся интерфейсы репозитория, отвечающие за взаимодействие с базой данных. Репозитории используют Spring Data JPA и содержат аннотацию `@Repository`. Они предоставляют методы для выполнения операций CRUD (создание, чтение, обновление, удаление) и других запросов к базе данных;
- Модуль сервисов (Service Module). Этот модуль содержит бизнес-логику приложения, которая обрабатывает запросы, полученные от контроллеров. Здесь находятся классы сервисов с аннотацией `@Service`, которые выполняют определенные операции и взаимодействуют с репозиториями.

Кроме этих основных модулей в приложении используются такие паттерны, как:

- DTO (Data Transfer Object) – компонент в архитектуре приложения, который используется для обмена данными между клиентской и серверной частями приложения. DTO представляют основные сущности или объекты приложения. DTO пересылаются между клиентской и серверной частями приложения в формате JSON.

База данных была развернута на хостинге bit.io. После создания базы данных были предоставлены данные для подключения, такие как хост, порт, имя базы данных, имя пользователя и пароль. Эти данные будут использоваться для настройки подключения к базе данных из приложения.

Сервер был развернут на хостинге Render. Этот процесс был выполнен с использованием системы контроля версий Git и Dockerfile. Render будет автоматически обнаруживать изменения и запускать процесс развертывания сервера. После загрузки потребовалось настроить переменные окружения, представляющие собой конфиденциальные настройки, такие как данные аутентификации для базы данных.

Для описания спецификации API использовался Swagger. Для интеграции его в проект потребовалось добавить соответствующую зависимость (springdoc-openapi) и аннотации к контроллерам, методам, моделям. На основе этих аннотаций и структуры приложения составляется интерактивная документация API.

4.3 Реализация Frontend

Клиентская часть приложения (frontend) разработана на языке JavaScript с использованием Фреймворка React.js. Для взаимодействия с серверной частью приложения на клиентской части используется библиотека axios, которая предоставляет интерфейс для выполнения HTTP-запросов из браузера.

Разрабатываемое приложение на React представляет собой SPA (single page application), то есть одностраничное приложение. Приложение загружает одну HTML страницу и динамически обновляет её без необходимости перезагрузки страницы при взаимодействии пользователя с приложением. Вместо традиционного подхода, когда каждый переход на новую страницу вызывает полную загрузку HTML, CSS и JavaScript, SPA загружает и инициализирует приложение один раз, а затем динамически обновляет только необходимую часть страницы при переходе между разделами или выполнении определенных действий.

Особенностью разработки приложения на React является разбиение на компоненты. Все приложение разбито на компоненты, которые в необходимый момент загружаются на страницу. Компоненты представляют собой функции, возвращающие определённый HTML код. Эти функции могут хранить в себе переменные или другие функции.

Разработанное React-приложение имеет базовую структуру: директорию public, в которой хранятся статические ресурсы (HTML файлы), и директорию src (source), в которой хранятся все исходные файлы приложения. Директория

src хранит в себе поддиректорию components (для хранения всех компонентов приложения), css (для хранения всех CSS файлов, описывающих стили), images (в ней хранятся необходимые приложению изображения) и store (в этой директории находится хранилище состояния приложения). Кроме того, в директории src находятся основные .js файлы: index.js и App.js.

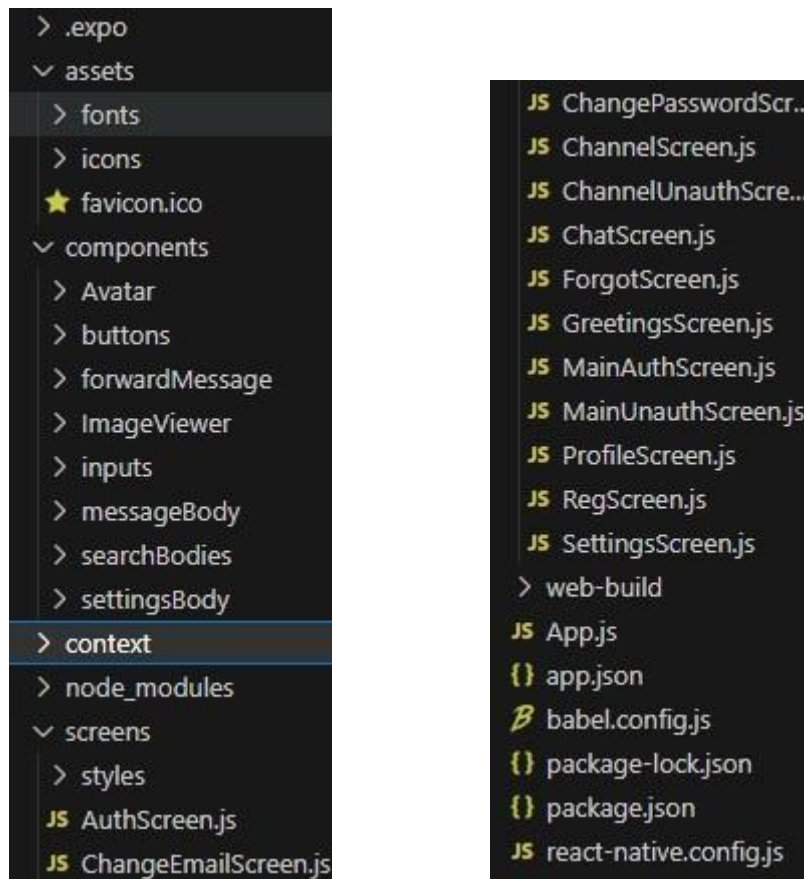


Рисунок 31 - Структура React-Native приложения

Основной HTML файл называется index.html, именно на него загружаются компоненты, которые будут выведены в браузере. В нём вызывается App.js, который является основным компонентом React-Native приложения. Он будет меняться в зависимости от действий пользователя. Все остальные компоненты вызываются по мере необходимости.

Компоненты могут состоять из других компонентов. Поэтому в приложении помимо компонентов, описывающих конкретные страницы, есть компоненты, из которых состоят эти страницы.

Так, среди файлов, обозначающих страницы, можно выделить:

- AuthScreen.js;
- ChangeEmailScreen.js;
- ChangePasswordScreen.js;
- ChannelScreen.js;
- ChannelUnauthScreen.js;
- ChatScreen.js;
- ForgotScreen.js;
- GreetingsScreen.js;
- MainAuthScreen.js;
- MainUnauthScreen.js;
- ProfileScreen.js;
- RegScreen.js;
- SettingsScreen.js;

Остальные компоненты являются частями страниц приложения.

Приложение имеет некоторые глобальные переменные, которые могут понадобиться в различных частях приложения, например, данные об авторизованном пользователе необходимы для отображения его профиля. Для хранения этих переменных и для их использования в приложении создано хранилище (Asyncstorage), в котором хранится состояние пользователя и состояние приложения. Оно было сделано при помощи Context. Context - это встроенный API, появившийся в React 16.3. Он позволяет передавать данные от родителя к дочерним элементам, вложенным глубоко в дерево компонентов, напрямую, вместо того чтобы передавать их через цепочку реквизитов.

Хранилище находится в директории context и называется AuthContext.js.

Он хранит себе состояние информации пользователя и состояние самого приложения (вошел ли пользователь в аккаунт).

4.4 Тестирование

Для тестирования разработанной системы были проведены тесты основных типов:

- Модульное тестирование (unit-тесты);
- Тесты, связанные с изменениями (дымовое тестирование);
- Функциональные тесты (GUI-тестирование).

4.4.1 Модульное тестирование

Unit-тесты являются частью процесса разработки программного обеспечения и используются для проверки отдельных модулей или компонентов программы. Они предназначены для тестирования функциональности отдельных единиц кода, таких как отдельные классы, методы или функции.

Целью unit-тестов является проверка, что каждая единица программного кода работает правильно в изоляции от других компонентов системы. Unit-тесты проверяют, что методы возвращают ожидаемые результаты при заданных входных данных, а также что ошибочные ситуации обрабатываются корректно.

На рисунке 32 представлены результаты unit-тестов серверной части приложения. Проверялась работа контроллеров. Тесты были написаны в классе с аннотацией @WebMvcTest.

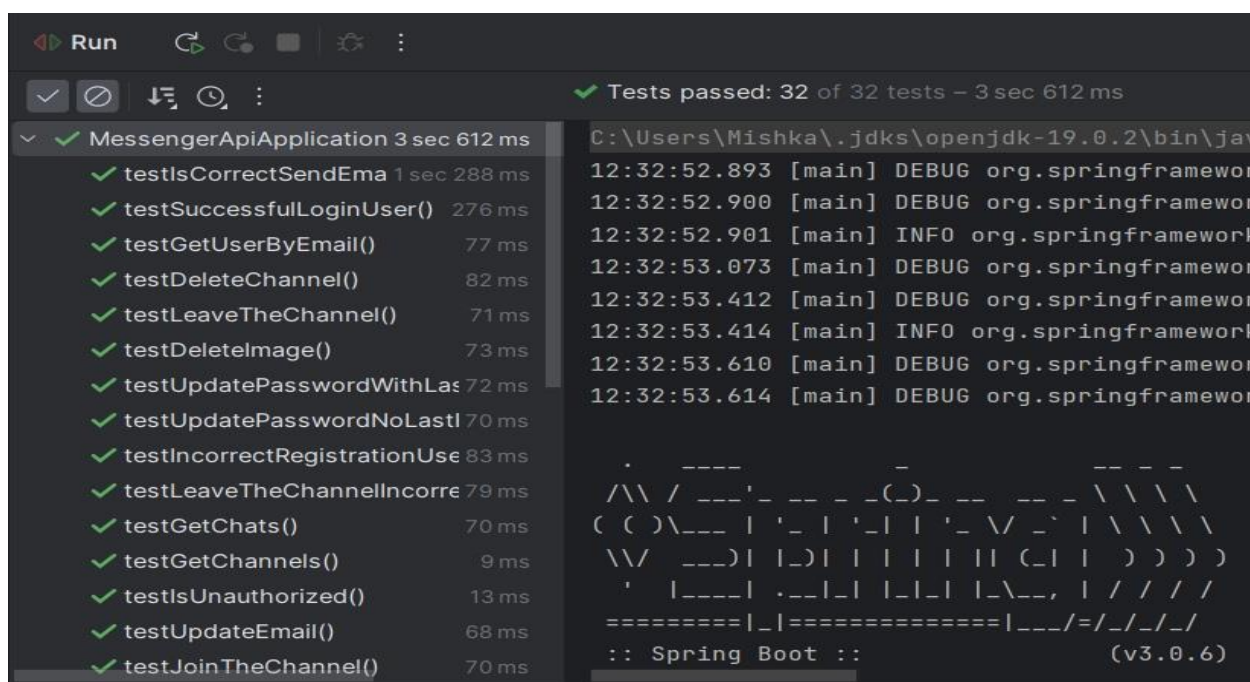


Рисунок 32 - Результаты unit-тестов

4.4.2 Дымовое тестирование

Дымовое тестирование (smoke testing) – это вид тестирования программного обеспечения, который выполняется для быстрой проверки основной функциональности системы или компонента после внесения изменений или внедрения новой версии. Основная цель дымового тестирования – обнаружить критические ошибки или проблемы, которые могут повлиять на работоспособность системы в целом.

В ходе дымового тестирования выполняются базовые операции или сценарии, которые предполагаются как наиболее важные и часто используемые пользователем.

В таблицах 1, 2, 3 представлены результаты дымового тестирования для основных сценариев клиента, администратора и создателя канала.

Таблица 1 - Результаты дымового тестирования для клиента

Тестовый сценарий	Результат теста
Регистрация	Пройден

Авторизация	Пройден
Просмотр каналов	Пройден
Просмотр каналов без авторизации аккаунта	Пройден
Просмотр чатов	Пройден
Поиск каналов и чатов	Пройден
Просмотр сохраненных сообщений	Пройден
Сохранение сообщений	Пройден
Создание чатов	Пройден
Создание каналов	Пройден
Изменение почты	Пройден
Изменение пароля	Пройден
Изменение пароля без авторизации аккаунта	Пройден
Изменение имени профиля	Пройден
Изменение фотографии профиля	Пройден
Изменение имени канала	Пройден
Изменение роли	Пройден
Отправка сообщений в чат	Пройден
Отправка сообщений в канал	Пройден
Удаление всех сохраненных сообщений	Пройден
Удаление конкретного сохраненного сообщения	Пройден
Присоединение к каналу	Пройден
Выход из канала	Пройден
Удаление чата	Пройден
Удаление канала	Пройден
Удаление участников канала	Пройден

Таблица 2 - Результаты дымового тестирования для администратора канала

Тестовый сценарий	Результат теста
Изменение названия канала	Пройден

Создание ролей для участников (с ролью не администратор) канала	Пройден
Исключение участников (с ролью не администратор) из канала	Пройден

Таблица 3 - Результаты дымового тестирования для создателя канала

Тестовый сценарий	Результат теста
Изменение названия канала	Пройден
Создание ролей для участников (любых) канала	Пройден
Исключение участников (любых) из канала	Пройден
Удаление каналов	Пройден

4.4.3 GUI-тестирование

Тестирование пользовательского интерфейса (GUI-тестирование) – это процесс тестирования элементов управления в приложении, который помогает убедиться, что интерфейс соответствует ожидаемой функциональности. Задача проведения GUI-тестов – убедиться, что в функциях пользовательского интерфейса отсутствуют дефекты.

В таблице 4, 5 представлены результаты тестирования пользовательского интерфейса. В ней отражены тестовые сценарии для авторизованного и неавторизованного пользователя.

Таблица 4 - Результаты GUI-тестирования для авторизованного пользователя

Тестовый сценарий	Ожидаемый результат	Статус теста
Нажатие на кнопку «Мой аккаунт» на странице авторизованного пользователя	Переход на страницу редактирование профиля	Пройден

Нажатие на кнопку «Изменить почту» на странице редактирования профиля	Переход на страницу изменения почты	Пройден
Нажатие на кнопку «Изменить пароль» на странице редактирования профиля	Переход на страницу изменения пароля	Пройден
Попытка изменения почты с незаполненным полем	Вывод предупреждения «Неверный формат почты»	Пройден
Попытка изменения пароля с незаполненным полем	Вывод предупреждения «Неверный формат пароля»	Пройден
Нажатие на «Поиск»	Переход на страницу чата или канала для авторизованного пользователя	Пройден
Нажатие на кнопку «Присоединиться»	Переход к интерфейсу участника канала	Пройден
Нажатие на кнопку «Настройки канала»	Переход на страницу настроек канала	Пройден
Нажатие на кнопку «Удалить канал»	Удаление канала и переход на главную страницу	Пройден
Нажатие на кнопку «Изменить имя канала»	Изменение имя канала	Пройден
Нажатие на кнопку «Удалить участника»	Удаление участника канала	Пройден

Нажатие на кнопку «Изменить роль»	Изменение роли и назначение прав администратора	Пройден
Нажатие на кнопку «Удалить чат»	Удаление чата	Пройден
Нажатие на кнопку «Удалить» на сохраненном сообщении	Удаление конкретного сохраненного сообщения	Пройден
Нажатие на кнопку «Удалить все» в сохраненных сообщениях	Удаление всех сохраненных сообщений	Пройден

Таблица 5 - Результаты GUI-тестирования для неавторизованного пользователя

Тестовый сценарий	Ожидаемый результат	Статус теста
Нажатие на кнопку «Зарегистрироваться»	Переход на страницу регистрация	Пройден
Нажатие на кнопку «Начать»	Переход на главную страницу для неавторизованного пользователя	Пройден
Нажатие на кнопку «Войти»	Переход на страницу входа в аккаунт	Пройден
Нажатие на кнопку «Создать канал»	Переход на страницу входа в аккаунт	Пройден
Нажатие на «Поиск»	Переход на страницу канала для неавторизованного пользователя	Пройден

Нажатие на кнопку «Присоединиться»	Переход на страницу входа в аккаунт	Пройден
Нажатие на кнопку «Логотип»	Переход на главную страницу неавторизованного пользователя	Пройден
Нажатие на кнопку «Забыли пароль»	Переход на страницу восстановление пароля	Пройден
Попытка входа с незаполненным полем для адреса электронной почты	Вывод предупреждения «Неверный формат почты»	Пройден
Попытка входа с незаполненным полем для пароля	Вывод предупреждения «Неверный формат пароля»	Пройден
Попытка регистрации с незаполненным полем для адреса электронной почты	Вывод предупреждения «Неверный формат почты»	Пройден
Попытка регистрации с незаполненным полем для пароля	Вывод предупреждения «Неверный формат пароля»	Пройден

Заключение

В результате курсовой работе был разработан веб-мессенджер «Linking», реализующее следующие функции:

1. Организация обмена сообщениями;
2. Возможность просмотра истории сообщений канала без регистрации;
3. Создание чатов/каналов;
4. Возможность добавить важные сообщения в раздел «Избранное» в один клик.
5. Кастомизация канала (добавление различных ролей для участников)

Разработанный мессенджер в полной мере соответствует требованиям, предъявленным на этапе постановки задачи. В ходе выполнения курсовой работы был проведен детальный анализ предметной области, а также технологий реализации и необходимых программных платформ, а также выполнены все этапы проектирования и разработки веб-мессенджера.

Список использованных источников

1. Rest, что это такое? [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/590679/>. – Заглавие с экрана. – (Дата обращения: 24.04.2023).
2. UML.indd [Электронный ресурс]. – Режим доступа: <https://csharpcooking.github.io/theory/UnifiedModelingLanguageUserGuide.pdf>. – Заглавие с экрана. – (Дата обращения: 27.04.2023).
3. Тестирование API с помощью Swagger: особенности и преимущества [Электронный ресурс]. – Режим доступа: <https://blog.ithillel.ua/ru/articles/api/testing-with-swagger>. – Заглавие с экрана. – (Дата обращения: 27.04.2023).
4. Render review 2023 [Электронный ресурс]. – Режим доступа: <https://www.websiteplanet.com/web-hosting/render/>. – Заглавие с экрана. – (Дата обращения: 25.04.2023).
5. Система управления объектно-реляционными базами данных PostgreSQL [Электронный ресурс]. – Режим доступа: <https://webcreator.ru/technologies/webdev/postgresql>. – Заглавие с экрана. – (Дата обращения: 20.04.2023).
6. Spring Framework Documentation [Электронный ресурс]. – Режим доступа: <https://docs.spring.io/spring-framework/reference/>. – Заглавие с экрана. – (Дата обращения: 24.05.2023).
7. Introduction - React-Native [Электронный ресурс]. – Режим доступа: <https://reactnative.dev/docs/getting-started>. – Заглавие с экрана. – (Дата обращения: 27.05.2023).
8. Vercel: The versatile platform for modern web development [Электронный ресурс]. – Режим доступа: <https://ikius.com/blog/what-is-vercel>. – Заглавие с экрана. – (Дата обращения: 29.05.2023).