

Министерство науки и высшего образования Российской Федерации  
**Муромский институт (филиал)**  
федерального государственного бюджетного образовательного учреждения высшего образования  
**«Владимирский государственный университет**  
**Имени Александра Григорьевича и Николая Григорьевича Столетовых»**  
**(МИВлГУ)**

Факультет ИТР

Кафедра ПИн

УТВЕРЖДАЮ

Зав. кафедрой

А.Л. Жизняков

(подпись)

«   » 2023г

# *БАКАЛАВРСКАЯ РАБОТА*

Тема Разработка web-сайта «Социальная сеть фотографов»  
МИВУ.09.03.04-8.000 ВКР

Руководитель

Быков А.А

(фамилия, инициалы)

(подпись)

(дата)

Студент ПИн-119

(группа)

Лямина И.А.

(фамилия, инициалы)

(подпись)

(дата)

Муром 2023

Задание

Задание

Темой выпускной квалификационной работы является разработка web-сайта «Социальная сеть для фотографов». В ходе выполнения выпускной квалификационной работы выявлены требования к программе, разработаны модели данных и диаграммы. На основе разработанных моделей создана БД в СУБД Microsoft SQL Server и разработан сервер на Web API ASP.NET Core, который работает с БД, и два типа клиентов на ASP.NET Core MVC: для пользователя и для сотрудника. Информационная система разрабатывалась в среде разработки Visual Studio 2022 на языке программирования C#. На заключительном этапе работы произведено тестирование разработанного продукта.

The theme of the final qualifying work is the development of the website "Social network for photographers". During the completion of the final qualification work, the requirements for the program were identified, data models and diagrams were developed. Based on the developed models, a database was created in the Microsoft SQL Server DBMS and a server was developed on the ASP.NET Core Web API that works with the database, and two types of clients on the ASP.NET Core MVC: for the user and for the employee. The information system was developed in the Visual Studio 2022 development environment in the C# programming language. At the final stage of the work, the developed product was tested.

## Содержание

Введение.....	6
1 Анализ технического задания .....	8
1.1 Аналоги .....	8
1.2 Требования к системе .....	14
1.3 Обоснование выбора средств реализации .....	16
2 Проектирование архитектуры программного продукта .....	18
2.1 Диаграмма вариантов использования .....	18
2.2 Функциональные диаграммы.....	20
2.3 Диаграммы потоков данных .....	29
2.4 Разработка моделей данных.....	38
3 Реализация программного продукта .....	42
3.1 Выбор и обоснование алгоритмов.....	42
3.2 Руководство программиста .....	44
3.3 Руководство пользователя.....	58
4 Тестирование программного продукта .....	64
Заключение .....	76
Список используемых источников.....	77
Приложение А. Снимки окон программы .....	78
Приложение Б. Исходный код ПО .....	90

Изм.	Лист	№ докум.	Подпись	Дата	МИВУ 09.03.04-8.000 ВКР		
Разраб.	Лямина И.А.						
Провер.	Быков А.А.						
Реценз.							
Н. Контр.	Быков А.А.						
Утврд.	Жизняков А.Л.						
Разработка web-сайта «Социальная сеть фотографов»					Лит.	Лист	Листов
						5	90
					МИ ВлГУ ПИн-119		

## Введение

Социальные сети – это интернет-площадки для общения, обмена информацией и контентом, прочих социальных взаимодействий. Они используются для работы, отдыха и развлечений, позволяют координировать между собой группы людей и имеют широкий набор функций. В отличие от более компактного формата мессенджера социальная сеть обычно поддерживает возможность выкладывать во всеобщий доступ и потреблять контент.

Социальные сети, с момента их появления все больше проникают в нашу жизнь. В настоящее время социальные сети становятся все более популярными, предоставляя пользователям возможность общаться, делиться информацией и находить новых друзей. Создание социальной сети для фотографов может быть особенно полезным, поскольку фотография — это одно из самых популярных средств для выражения своих мыслей и чувств в Интернете.

В данной бакалаврской работе рассматривается разработка web-приложения «Социальная сеть для фотографов». Цель работы - проектирование и создание удобного и функционального веб-приложения, которое позволит фотографам и обычным пользователям общаться, делиться своими работами и просматривать работы других пользователей.

Создание социальной сети для фотографов является актуальной задачей, так как фотографам необходимо осуществлять продвижение своей работы, создавать портфолио и следить за работами коллег. Также в связи с блокировкой Instagram в России исчезла самая востребованная социальная сеть, а большинство аналогов не предоставляют интерфейса на русском языке.

Результаты работы должны обеспечить возможность:

- хранить работы в одном месте (создание портфолио);
- продемонстрировать свои работы более широкой аудитории (продвижения фотобизнеса, возможность найти своих клиентов);

Изм.	Лист	№ докум.	Подпись	Дата

- общаться с другими пользователями;
- просматривать работы других пользователей.

Для достижения цели были поставлены следующие задачи:

- проанализировать предметную область;
- разработать модели данных;
- реализовать базу данных;
- разработать серверное приложение;
- разработать два типа клиентских приложений: для пользователя и для сотрудника/администратора;
- протестировать программный продукт.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

# 1 Анализ технического задания

## 1.1 Аналоги

Для наиболее точного анализа предоставленного технического задания были найдены несколько аналогов разрабатываемого программного средства.

Общие функции аналогов:

- авторизация и регистрация пользователей;
- просмотр и изменение личного профиля;
- размещение фотографии с подписью (пост);
- подписка (отписка) на других пользователей;
- возможность просмотра в профиле подписчиков и подписок;
- просмотр профилей других пользователей с постами, которые они разместили;
- просмотр новостей от фотографов, на которых подписан пользователь;
- комментирование размещённых постов (с просмотром всех предыдущих комментариев) и возможность ставить отметки нравится;
- поиск по людям и постам.

### 1.1.1 Flickr [2]

Функции:

- возможность создания альбомов;
- добавление понравившихся фото в избранное;
- добавление тегов к постам;
- создание блогов;
- поиск по блогам;
- добавление категорий для блогов.

Недостатки:

Изм.	Лист	№ докум.	Подпись	Дата

- нет фильтраций для постов;
- блоги – это как отдельный сайт;
- нет русского языка.

Примеры интерфейса приложения приведены на рисунках 1, 2 и 3.

Особенности интерфейса:

- список постов выводится как галерея фотографий, не видно логина фотографа и подписи к фото;
- есть отдельная страница с наиболее популярными тегами, а не просто поиск по тегам.

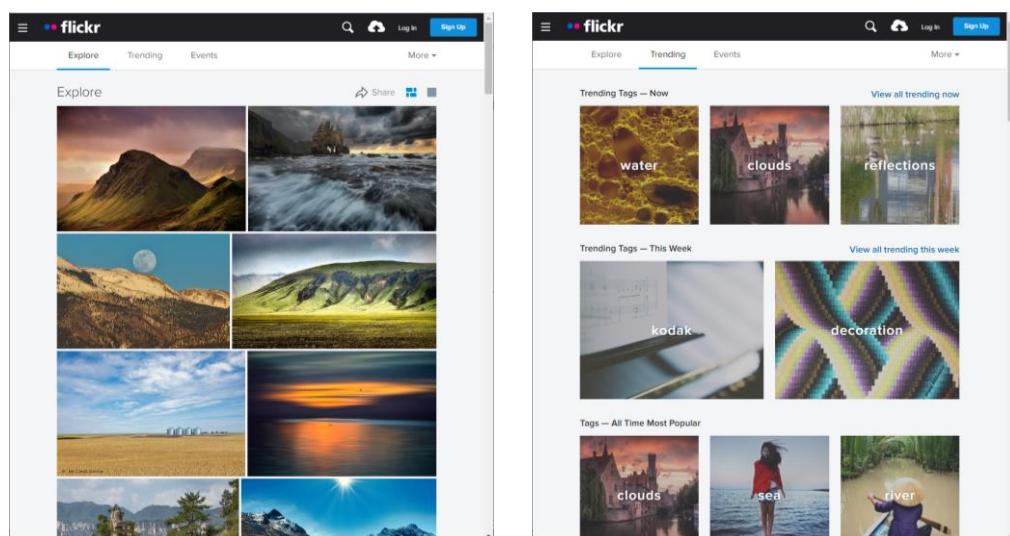


Рисунок 1 – скриншоты списка постов и фильтрации по тегам

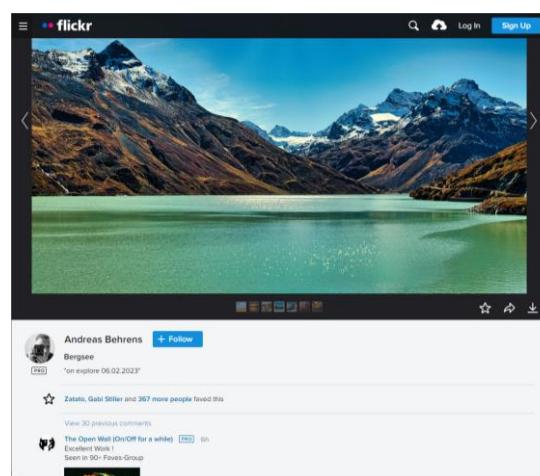


Рисунок 2 – скриншот поста

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

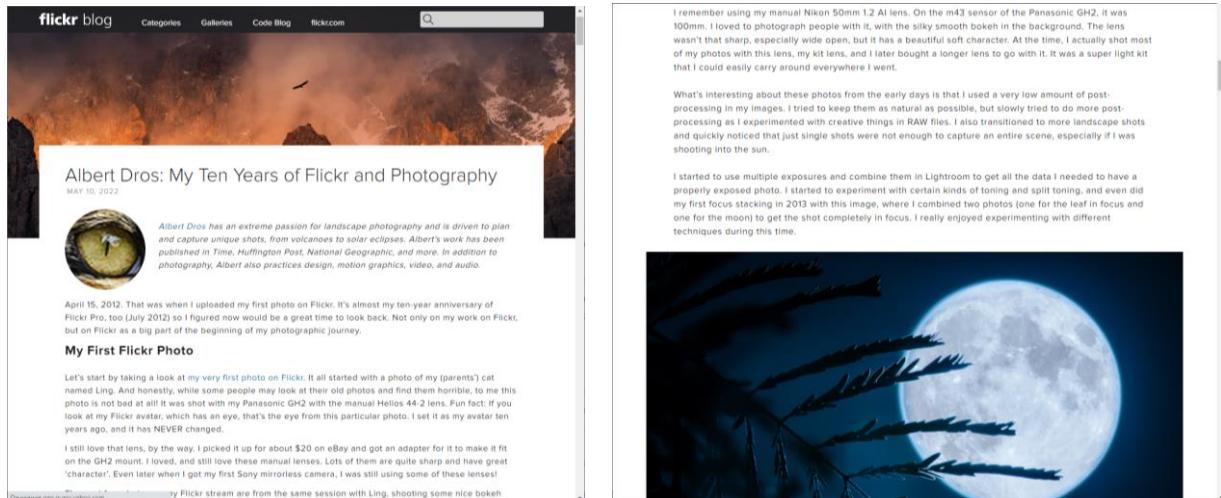


Рисунок 3 – скриншоты блога

### 1.1.2 500px [1]

#### Функции:

- добавление к посту характеристик фотоаппарата, на который было сделано фото;
- возможность добавления категорий к постам;
- фильтрация данных (по категориям и что показывать новые/популярные);
- блог от разработчиков;
- просмотр популярных постов и новинок.

Недостатки: нет русского языка.

Примеры интерфейса приложения приведены на рисунках 4 и 5.

#### Особенности интерфейса:

- список постов выводится как галерея фотографий, не видно логина фотографа и подписи к фото;
- на больших экранах комментарии к постам расположены в отдельной колонке, а не идут после всей остальной информации.

Изм.	Лист	№ докум.	Подпись	Дата

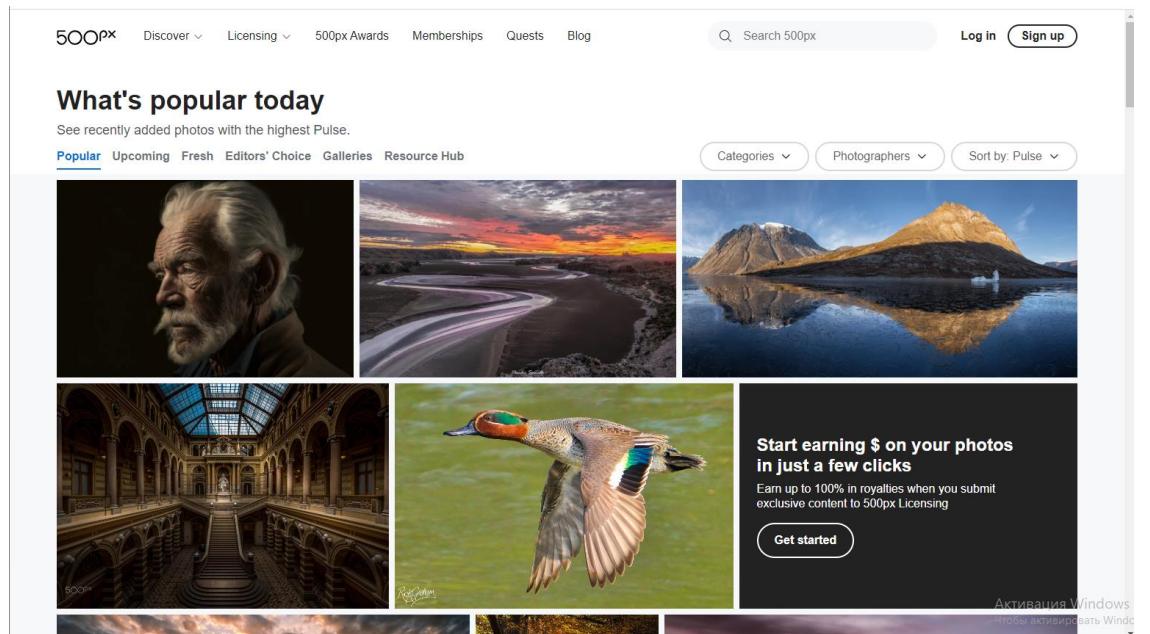


Рисунок 4 – скриншот списка постов

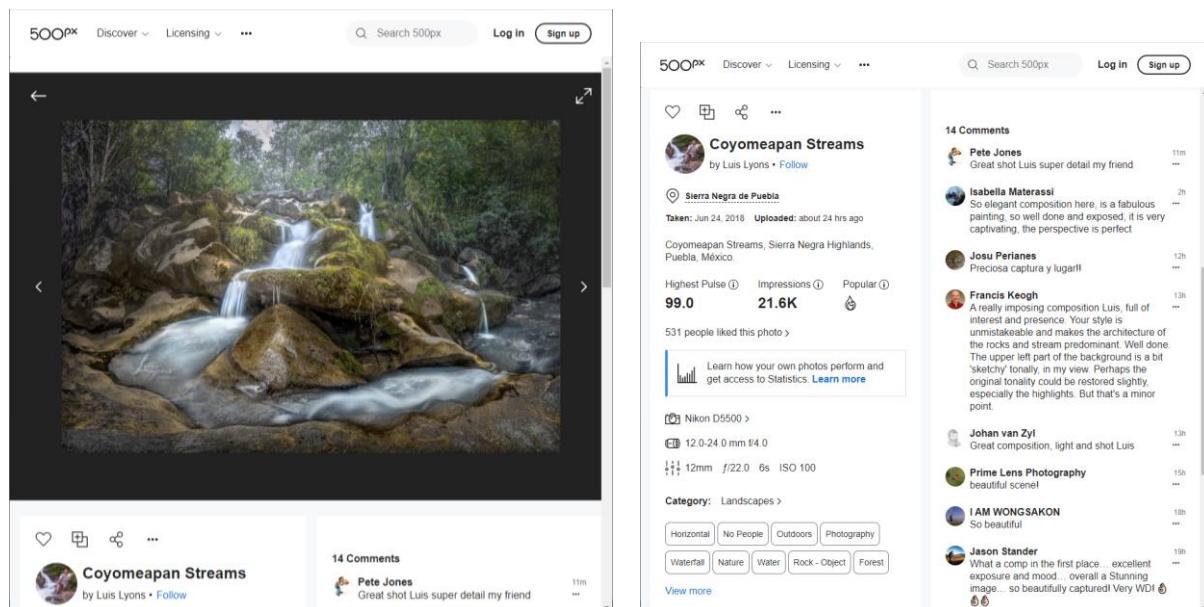


Рисунок 5 – скриншоты поста

### 1.1.3 Instagram (запрещён в России)

Функции:

- возможность делиться видеозаписями помимо фотографий;
- возможность добавление нескольких фотографий в посте;
- добавление тегов к постам.

Изм.	Лист	№ докум.	Подпись	Дата

МИВУ 09.03.04-8.000 ВКР

Лист  
11

Примеры интерфейса приложения приведены на рисунке 6.

### Особенности интерфейса:

- так как есть возможность добавлять в один пост несколько фотографий, то они объединяются в слайдер.

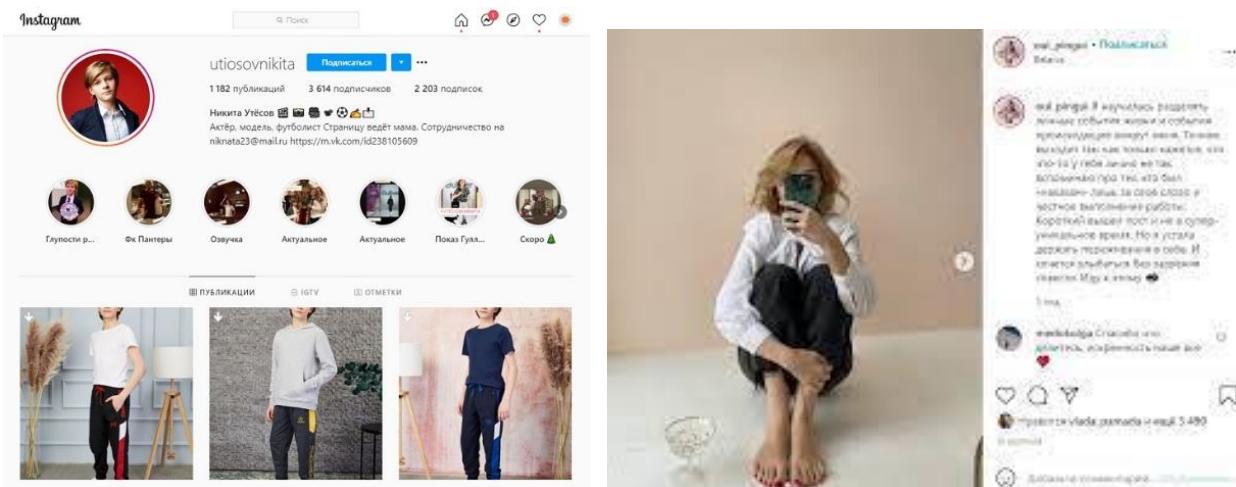


Рисунок 6 – скриншоты профиля пользователя и поста пользователя

### 1.1.4 Habr [3]

Общие функции не относятся к данному аналогу, аналог выбран для рассмотрения ведения блогов.

#### Функции:

- создание блогов;
- просмотр блогов других пользователей;
- фильтрация данных (что сначала показывать новые или лучшие, для новых дополнительно порог рейтинга, для лучших - период);
- возможность просмотра блогов по категориям;
- поиск по блогам;
- комментарии, отметка нравится, сохранение блога;
- добавление тегов к блогам;
- добавление нескольких категорий блогам.

Изм.	Лист	№ докум.	Подпись	Дата

Примеры интерфейса приложения приведены на рисунках 7 и 8.

### Особенности интерфейса:

- изображения следуют за текстом и занимают всю ширину блога.

Можно добавить только одно изображение (не слайдер);

- в блогах можно добавлять следующие компоненты: заголовок, цитата, список, нумерованный список, медиаэлемент, изображение, разделитель, таблица, код, формула, спойлер, якорь, персона.

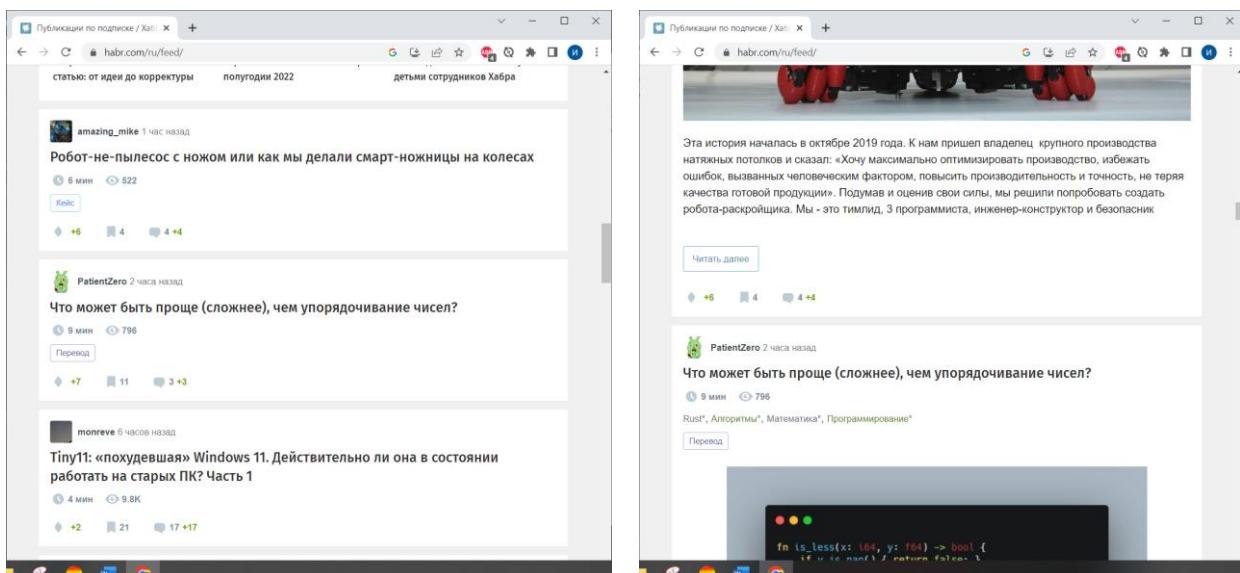


Рисунок 7 – скриншоты компактной и классической ленты с блогами

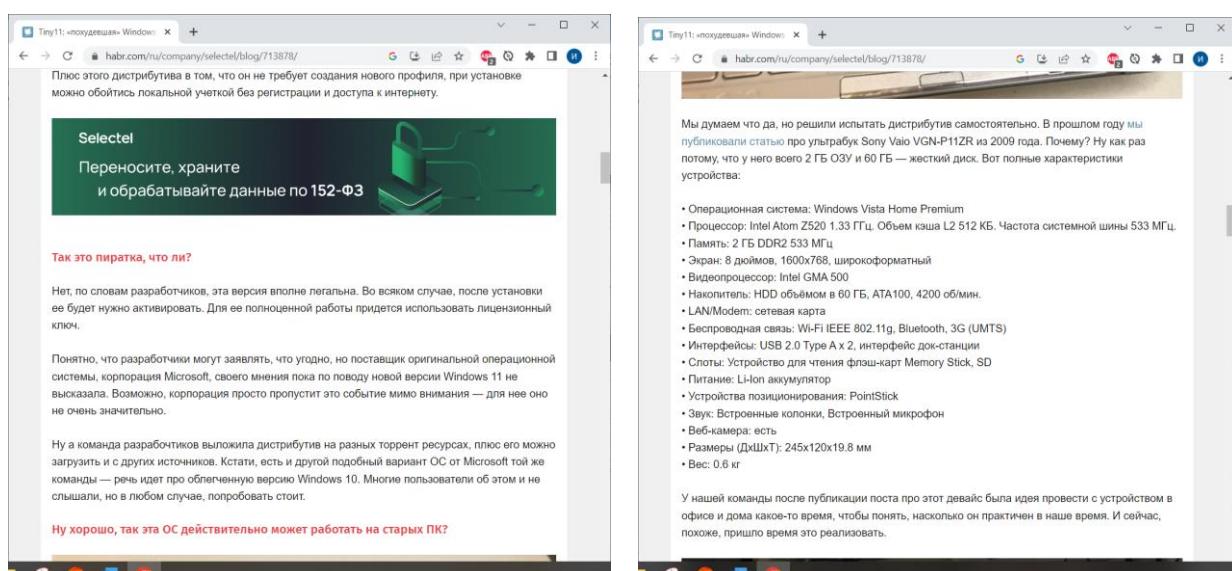


Рисунок 8 – скриншоты блога

Изм.	Лист	№ докум.	Подпись	Дата

## 1.2 Требования к системе

### 1.2.1 Функциональные требования

Разрабатываемый программный продукт рассчитан на довольно широкий круг людей. Целевая аудитория — это люди, увлеченные фотографией, независимо от уровня опыта и профессиональной подготовки. Это могут быть фотографы любители, студенты фотошкол, профессиональные фотографы и т.д. Они интересуются профессиональной сферой и желают повысить свои навыки и обменяться опытом с другими участниками.

Также программой будут пользоваться сотрудники и администраторы, основная задача которых — обработка жалоб от пользователей.

Программа должна содержать следующие функциональные возможности:

1. Для пользователя:
  - авторизация и регистрация пользователей;
  - просмотр постов/блогов (в том числе своих и сохранённых) и изменение личного профиля;
  - размещение постов и блогов;
  - выбор категорий при размещении поста/блога;
  - комментирование постов/блогов;
  - возможность ставить отметки нравится постам/блогам;
  - возможность сохранять в избранное посты/блоги;
  - просмотр профилей других пользователей с постами/блогами, которые они разместили;
  - подписка/отписка на/от других пользователей;
  - просмотр новых постов от пользователей, на которых была осуществлена подписка (лента новостей);
  - возможность подать жалобу на пост/блог другого пользователя;
  - фильтрация данных в ленте новостей,

Изм.	Лист	№ докум.	Подпись	Дата

– поиск пользователей, постов и блогов.

2. Для сотрудника и администратора:

– авторизация;

– просмотр своего профиля;

– просмотр жалоб на посты/блоги и возможность заблокировать данные посты.

3. Для администратора:

– работа со списком сотрудников, категорий, базовых жалоб: просмотр, добавление, изменение и удаление.

### 1.2.2 Нефункциональные требования

1. Требования к интерфейсу:

– пользователи и сотрудники будут иметь разный интерфейс и эти интерфейсы никак не должны пересекаться между собой;

– веб-страницы сайта должны быть адаптивные;

– на сайте должно быть меню для навигации;

– при авторизации перед пользователем появляется лента новостей, а перед сотрудником список постов/блогов, на которые была подана жалоба;

– неавторизованный пользователь не может работать с приложением.

2. Требования к данным:

Все данные хранятся в БД. С БД работает сервер, к которому для обработки и получения данных обращаются клиенты.

3. Требования к безопасности.

С приложением будут работать множество пользователей, поэтому для обеспечения конфиденциальности и распределения ролей вход на сайт будет осуществляться с помощью уникального логина и пароля. После успешной авторизации пользователю будет выдаваться токен, который будет использоваться для дальнейших запросов к серверу.

Изм.	Лист	№ докум.	Подпись	Дата

### 1.2.3 Эксплуатационные ограничения системы

Требования для компьютера, на котором стоит сервер:

- операционная система: Windows 10 TH1 1507 или более поздней версии;
- жесткий диск: минимум 6 ГБ свободного места на диске;
- память: для обеспечения оптимальной производительности требуется не менее 4 ГБ с последующим увеличением по мере роста размера базы данных;
- быстродействие процессора: минимум - процессор x64 с тактовой частотой 1,4 ГГц, рекомендуется - 2,0 ГГц и выше

Требования для клиентов: доступ к разработанному приложению для пользователей должен быть возможен при помощи стандартных WEB-обозревателей (минимальные версии: Internet Explorer 11, Firefox 60, Safari 12, Google Chrome 60, Opera 30, Yandex 21), а также с мобильных устройств, имеющих WEB-браузер.

### 1.3 Обоснование выбора средств реализации

Выбор стека технологий для разработки веб-приложения является важным этапом проектирования, который должен основываться на требованиях и целях разрабатываемой информационной системы.

Для разработки клиентских приложений была выбрана платформа ASP.NET C#, которая является одной из наиболее популярных для разработки веб-приложений. Она обладает широкими возможностями для создания динамических веб-страниц, обработки форм, доступа к данным и других функций, которые могут потребоваться в разработке.

Сервер для приложения будет написан с использованием Web API ASP.NET, которая предоставляет мощный механизм для обработки HTTP-запросов и формирования ответов, а также инструменты для авторизации,

Изм.	Лист	№ докум.	Подпись	Дата

аутентификации. Это позволяет создавать RESTful API, что важно для разработки современных веб-приложений.

В качестве СУБД была выбрана MS SQL Server. Она является одной из наиболее распространенных и популярных СУБД, которая обладает высокой производительностью, масштабируемостью и надежностью.

Также выбор технологий для сервера и СУБД связан с использованием Entity Framework (EF). EF — это ORM-фреймворк, который позволяет упростить работу с базами данных и ускорить процесс разработки веб-приложений. Он позволяет разработчикам работать с данными в объектно-ориентированной форме, а не в виде запросов на SQL-языке, что упрощает проектирование и реализацию приложения.

Web API ASP.NET имеет интеграцию с EF, что позволяет разработчикам создавать RESTful API для работы с данными в базе данных через EF. Это упрощает процесс создания API и позволяет сократить время разработки.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

## 2 Проектирование архитектуры программного продукта

Проектирование работы системы является важным этапом в разработке любой ИС, связанного с обработкой и хранением данных. Целью проектирования является создание информационной системы, которая будет максимально эффективна в использовании, а также будет соответствовать всем требованиям бизнес-процессов и задач, которые она должна решать.

### 2.1 Диаграмма вариантов использования

Создав диаграмму вариантов использования, которая представлена на рисунках 9 и 10, были сформулированы общие требования к функциональному поведению проектируемой системы и определены общие границы и контекст моделируемой предметной области.

Актерами в диаграмме являются: Пользователь, Сотрудник и Администратор. В самой диаграмме отображен план работы с ИС «Социальная сеть для фотографов». Пользователь, Сотрудник и Администратор – внешняя часть программы. Основной частью программы является:

- для пользователя: авторизация, просмотр ленты, поиск, взаимодействие с профилем, профилем другого пользователя, выбранным постом/блогом, выбранным постом/блогом другого пользователя;
- для сотрудника и администратора: авторизация, взаимодействие с постами/блогами, на которые была подана жалоба;
- для администратора: работа с сотрудниками, работа с категориями постов/блогов, работа с базовыми жалобами.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

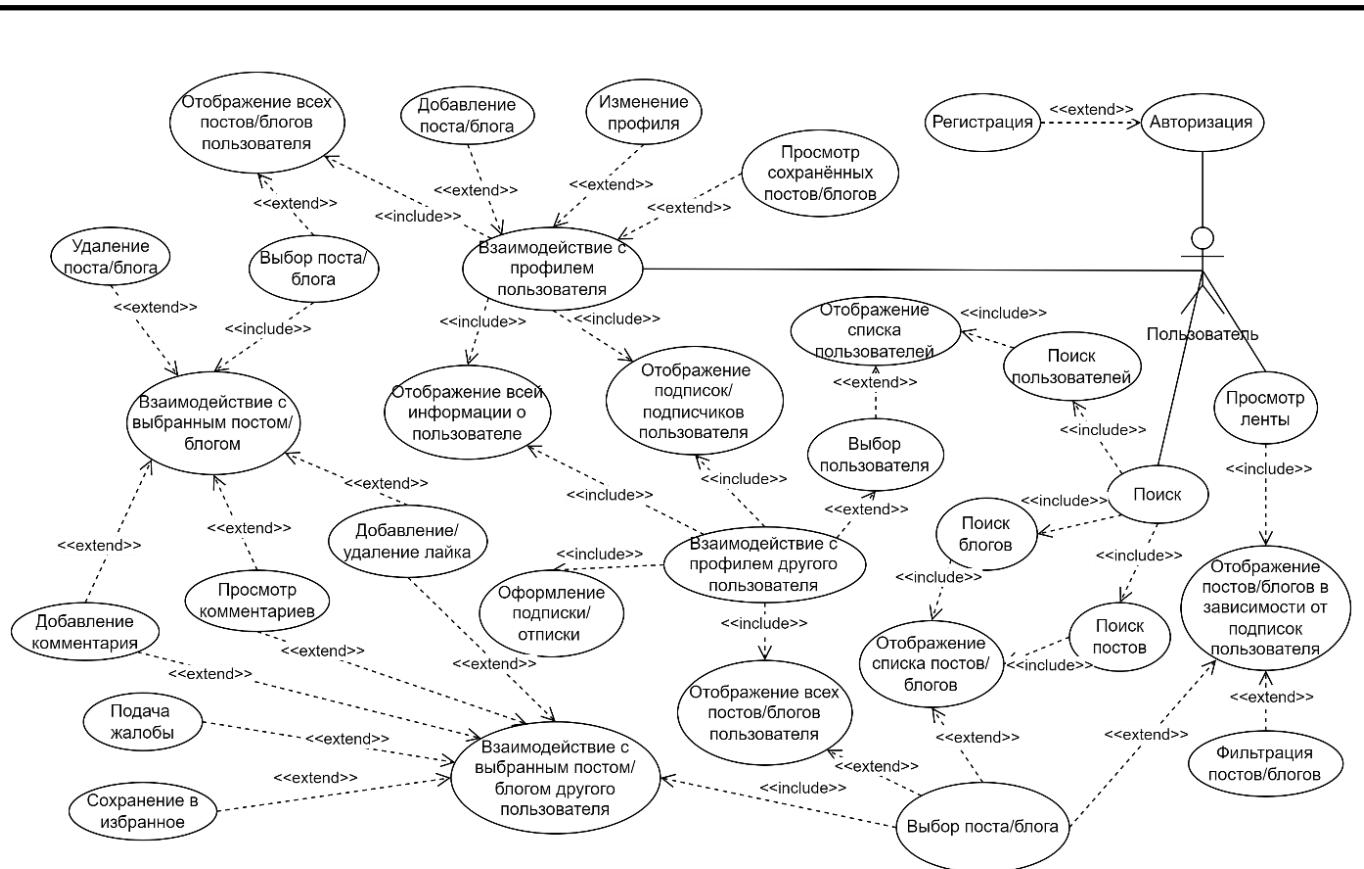


Рисунок 9 – диаграмма вариантов использования для пользователя

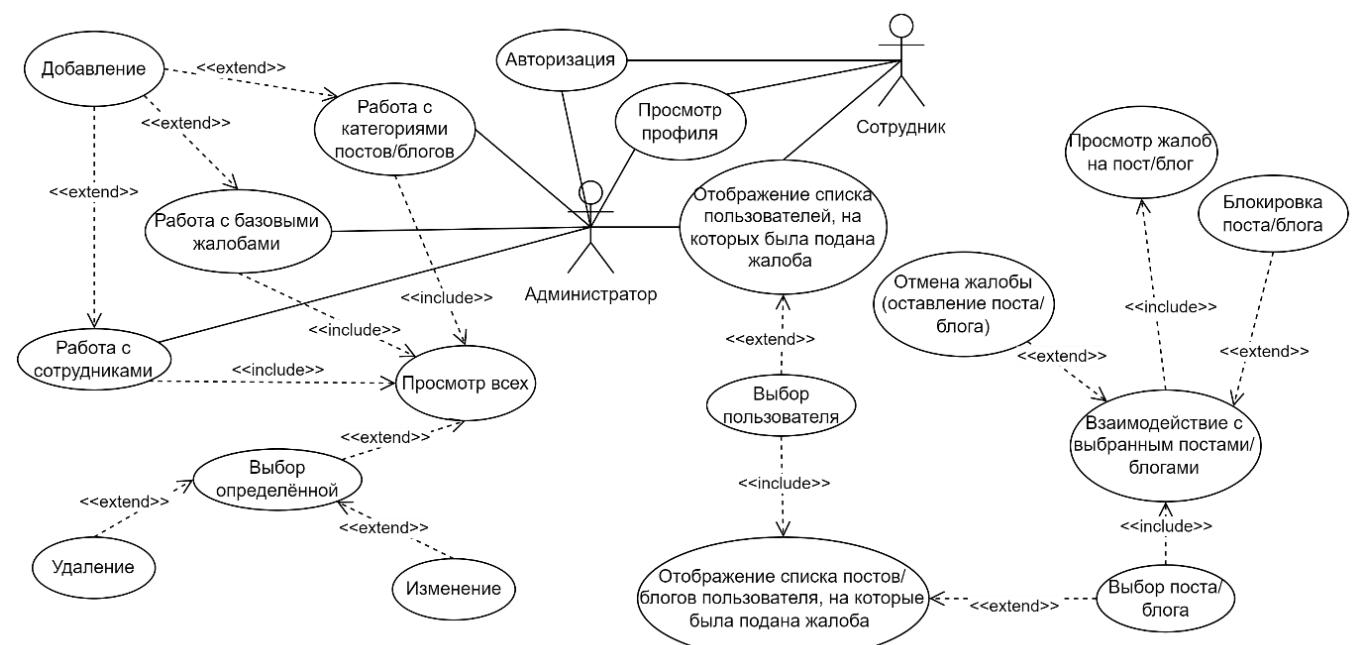


Рисунок 10 – диаграмма вариантов использования для  
сотрудника/администратора

<i>Изм.</i>	<i>Лист</i>	<i>№ докум.</i>	<i>Подпись</i>	<i>Дата</i>

## 2.2 Функциональные диаграммы

### 2.2.1 Функциональная диаграмма для клиента пользователя

Создадим функциональную диаграмму. Представим всю систему в виде простейшей компоненты – одного блока и дуг, изображающих интерфейсы с функциями вне системы. Созданная функциональная диаграмма показана на рисунке 11.

Информация, которая подвергается обработке (данные пользователя, новые данные для изменения профиля, данные для добавления поста/блога, выбранные фильтры, новые посты/блоги в зависимости от подписок, поисковые данные, найденные данные, текст комментария), входит с левой стороны блока, а результаты выхода (изменённые данные пользователя, данные нового поста/блога, изменённые данные подписок) показаны с правой стороны. Механизмы (пользователь), которые осуществляют операцию, представляются дугой, входящей в блок снизу.

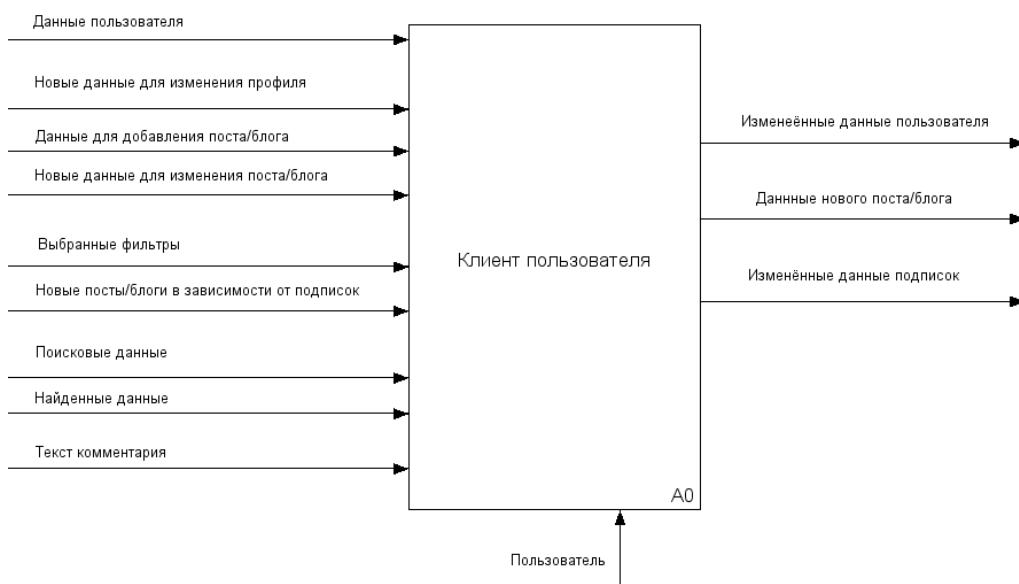


Рисунок 11 - диаграмма IDEF0 (для пользователя)

Изм.	Лист	№ докум.	Подпись	Дата

Для получения подробной информации о функционировании приложения для пользователя необходимо создать подробную функциональную диаграмму. Она показана на рисунке 12.

Пользователь передает данные для авторизации в системе. После авторизации при правильном вводе логина и пароля пользователь получает право доступа в приложение. Он может взаимодействовать со своим профилем, просматривать ленту или осуществлять поиск.

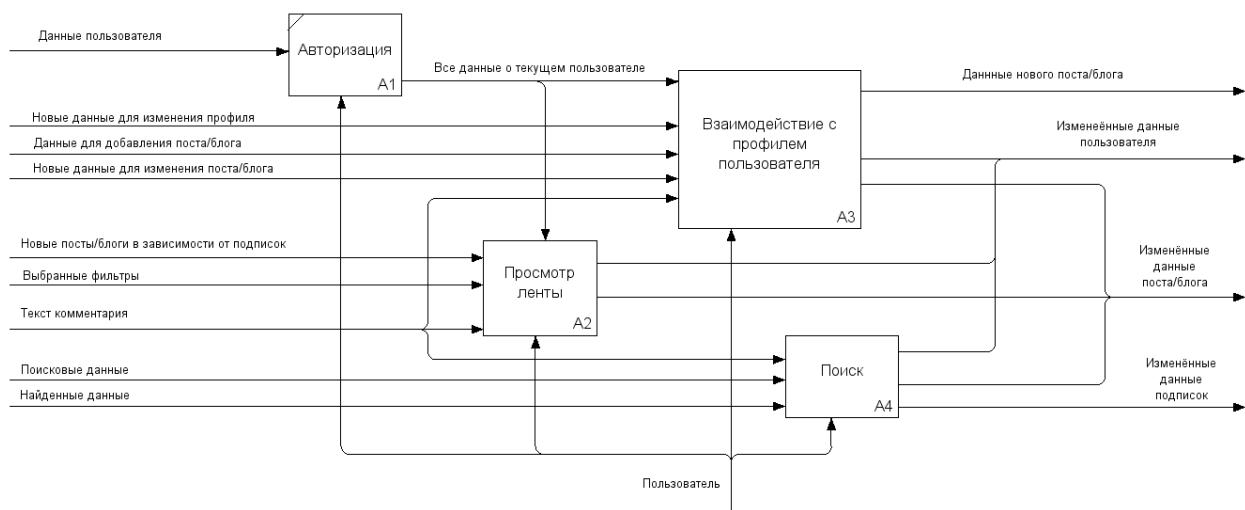


Рисунок 12 – Подробная диаграмма IDEF0 (для пользователя)

Подробная диаграмма на рисунке 13, раскрывает функционирование блока «Взаимодействие с профилем пользователя». При входе в профиль для пользователя отображается вся его информация, а также все его посты и блоги, с которыми он может взаимодействовать. Также пользователь может добавить новый пост или блог, изменить профиль или просмотреть все свои сохранённые посты/блоги с дальнейшим взаимодействием с ними.

Изм.	Лист	№ докум.	Подпись	Дата

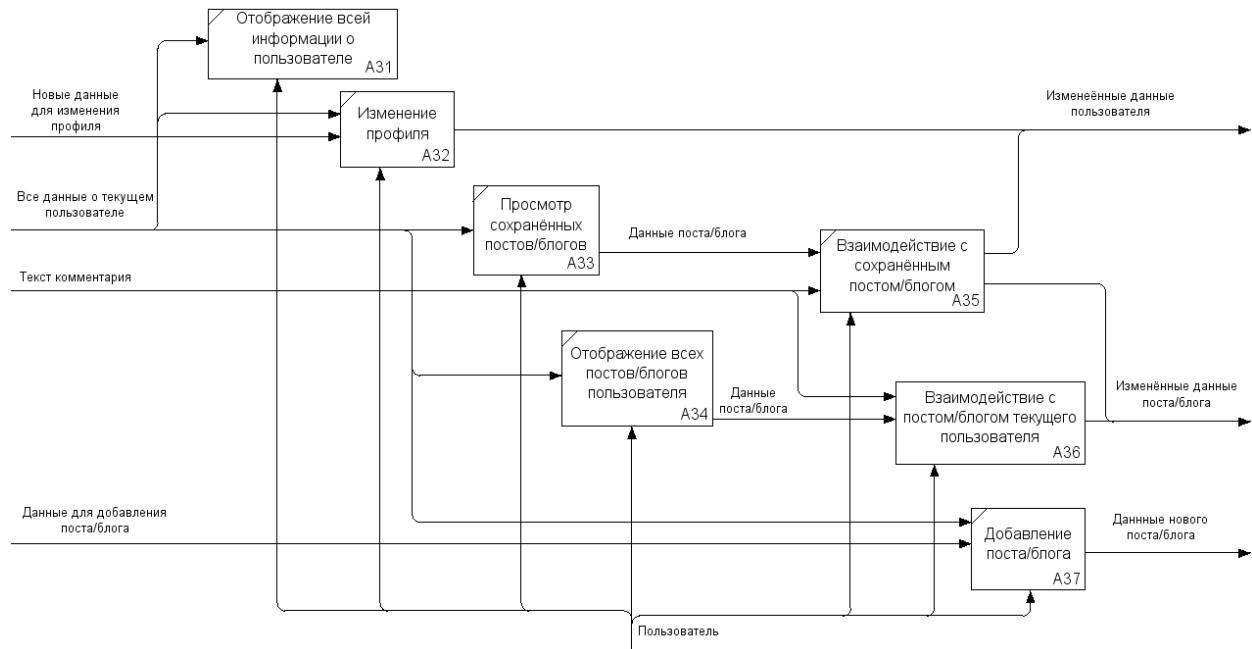


Рисунок 13 - Подробная диаграмма IDEF0 (Взаимодействие с профилем пользователя)

Функционирование блока «Взаимодействие с постом/блогом текущего пользователя» раскрыто на рисунке 14. Пользователь может просмотреть все комментарии и добавить свой. Также пользователь удалить пост/блог и добавить/удалить лайк.

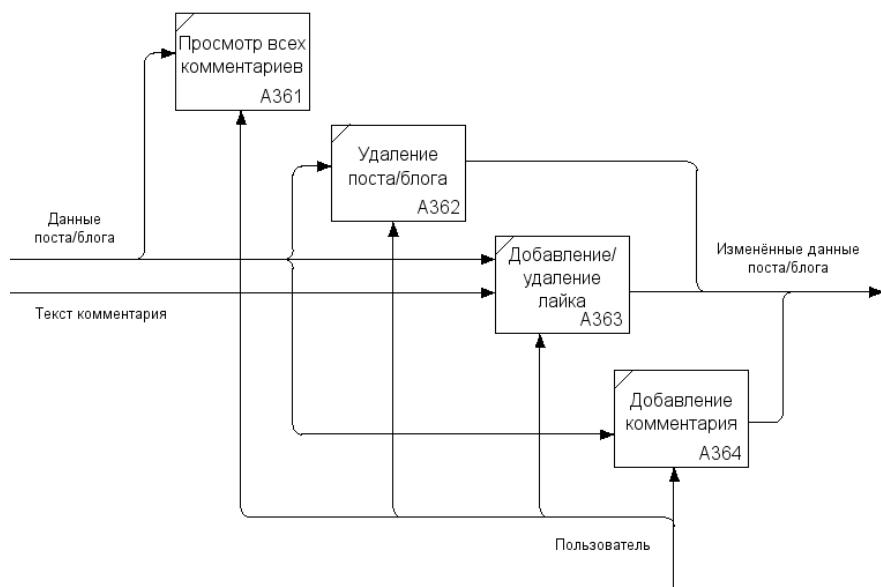


Рисунок 14 - Подробная диаграмма IDEF0 (Взаимодействие с постом/блогом текущего пользователя)

Изм.	Лист	№ докум.	Подпись	Дата

Подробная диаграмма «Поиск», которая показана на рисунке 15, раскрывает функционирование этого блока. Пользователь может осуществлять поиск по пользователям или постам/блогам с дальнейшей работой с найденной информацией.

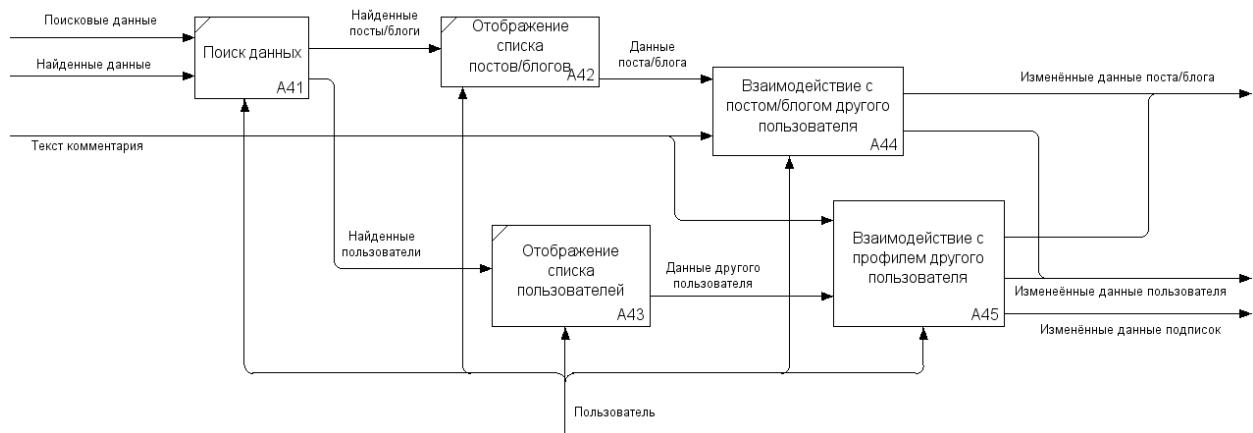


Рисунок 15 - Подробная диаграмма IDEF0 (Поиск)

Рисунок 16 показывает функционирование блока «Взаимодействие с профилем другого пользователя». Здесь отображается вся информация о пользователе и также можно оформить или удалить подписку на пользователя.

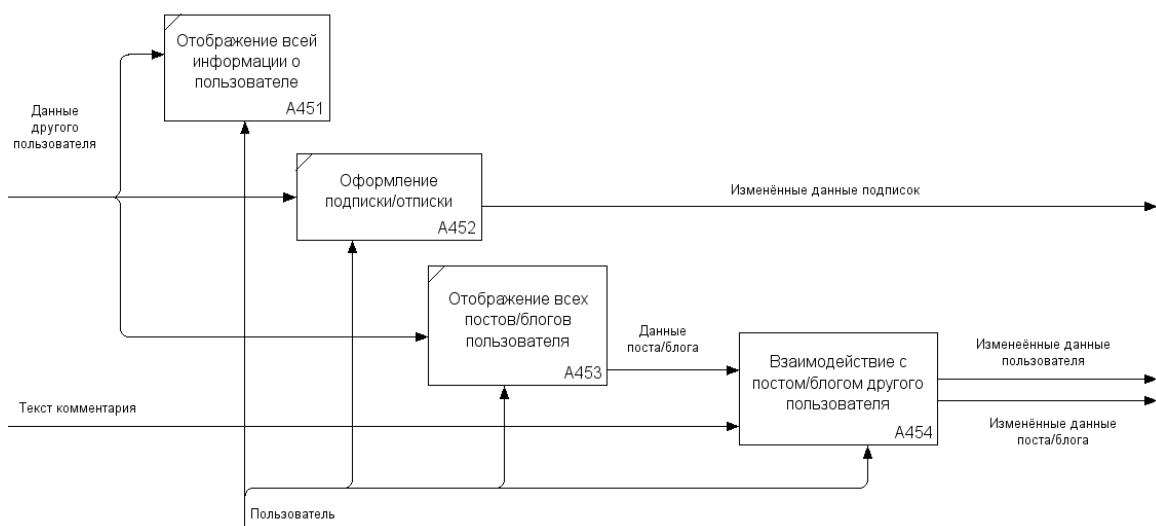


Рисунок 16 - Подробная диаграмма IDEF0 (Взаимодействие с профилем другого пользователя)

Подробная диаграмма на рисунке 17, раскрывает функционирование блока «Взаимодействие с постом/блогом другого пользователя». Пользователь может добавить/удалить лайк, просмотреть все комментарии и добавить свой, подать жалобу на пост/блог или сохранить в избранной.

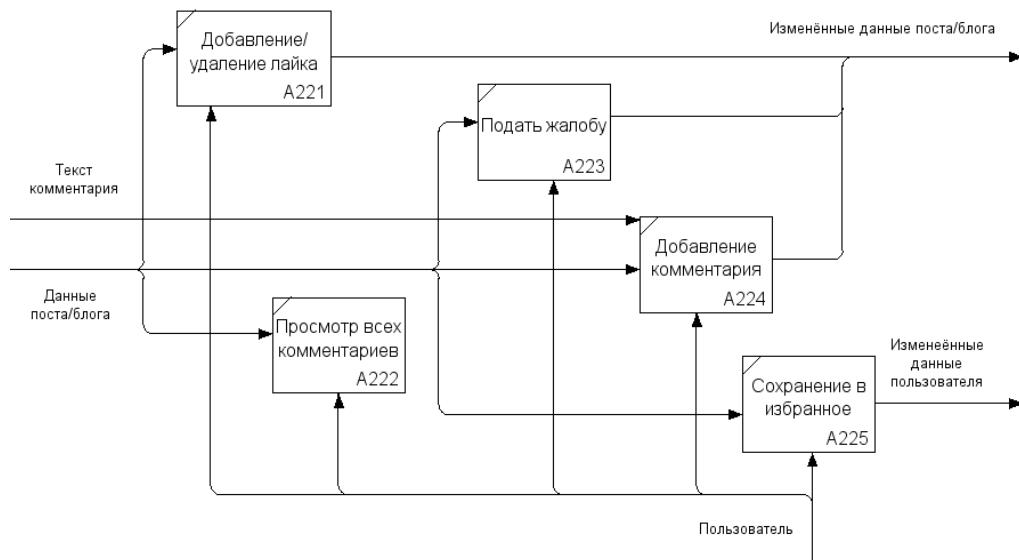


Рисунок 17 - Подробная диаграмма IDEF0 (Взаимодействие с постом/блогом другого пользователя)

Подробная диаграмма «Просмотр ленты», которая показана на рисунке 18, раскрывает функционирование этого блока. При открытии ленты новостей пользователю отображаются посты/блоги в зависимости от его подписок.

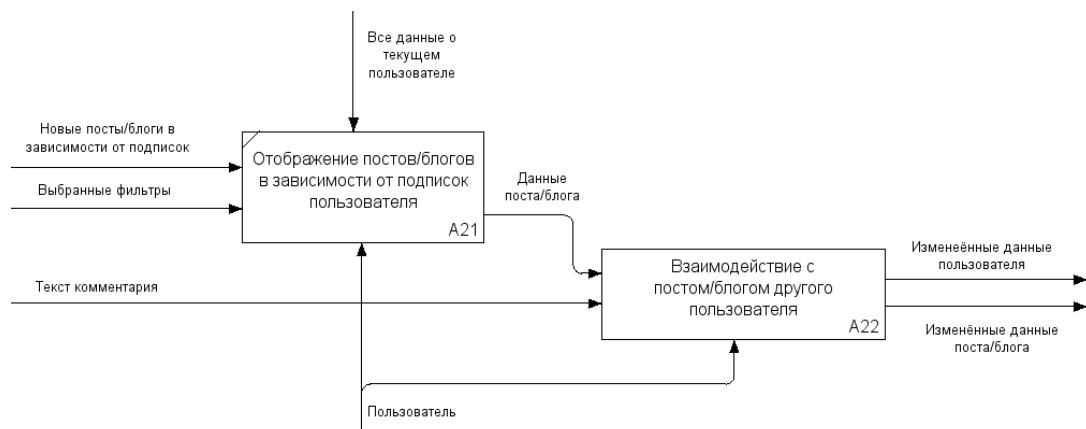


Рисунок 18 - Подробная диаграмма IDEF0 (Просмотр ленты)

Изм.	Лист	№ докум.	Подпись	Дата

## 2.2.2 Функциональная диаграмма для клиента сотрудника/администратора

Для сотрудников была создана отдельная функциональная диаграмма, представленная на рисунке 19.

Управляющая информацией здесь является решение проверяющего, и данная стрелка входит в блок сверху.

Информация, которая подвергается обработке (показана с левой стороны блока) - данные пользователя, список постов/блогов, на которые была подана жалоба, список базовых жалоб, данные для добавления базовой жалобы, новые данные для изменения базовой жалобы, список категорий постов/блогов, данные для добавления категории, новые данные для изменения категории, список сотрудников, данные для добавления сотрудника, новые данные для изменения сотрудника.

С правой стороны блока показаны результаты выхода - изменённые данные поста/блога, данные новой базовой жалобы, изменённые данные базовой жалобы, данные новой категории, изменённые данные категории, данные нового сотрудника, изменённые данные сотрудника.

Механизмы (сотрудник, администратор), которые осуществляют операцию, представляются дугой, входящей в блок снизу.

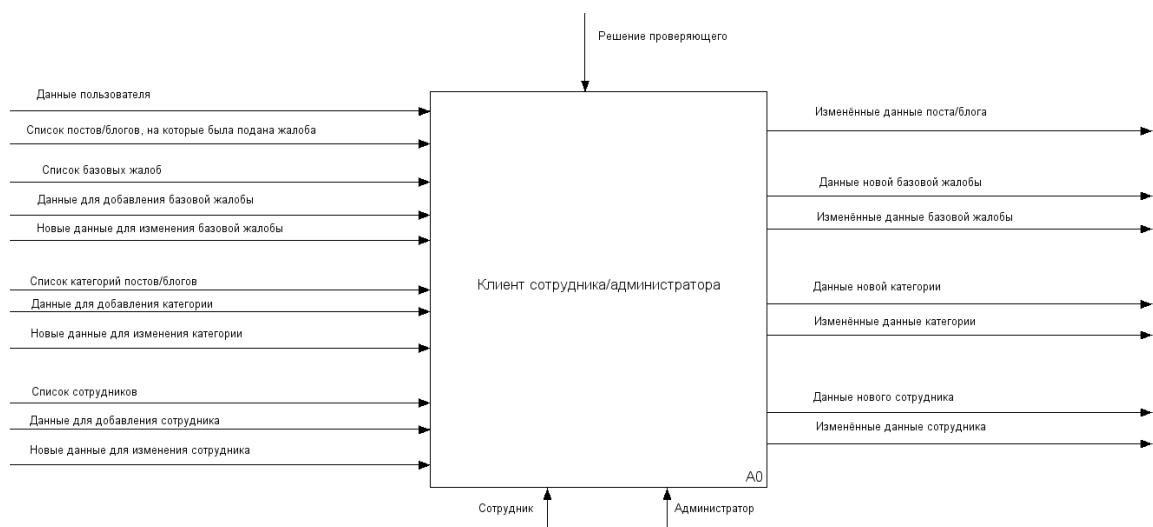


Рисунок 19 - Диаграмма IDEF0 (для сотрудника)

Изм.	Лист	№ докум.	Подпись	Дата

На рисунке 20 показана подробная функциональная диаграмма для приложения сотрудников. Пользователь передает данные для авторизации в системе после чего получает права доступа. Он может просматривать свой профиль, работать с базовыми жалобами, сотрудниками и категориями постов/блогов. Также пользователь может просматривать список постов/блогов, на которые была подана жалоба, и взаимодействовать с определённым компонентом данного списка.

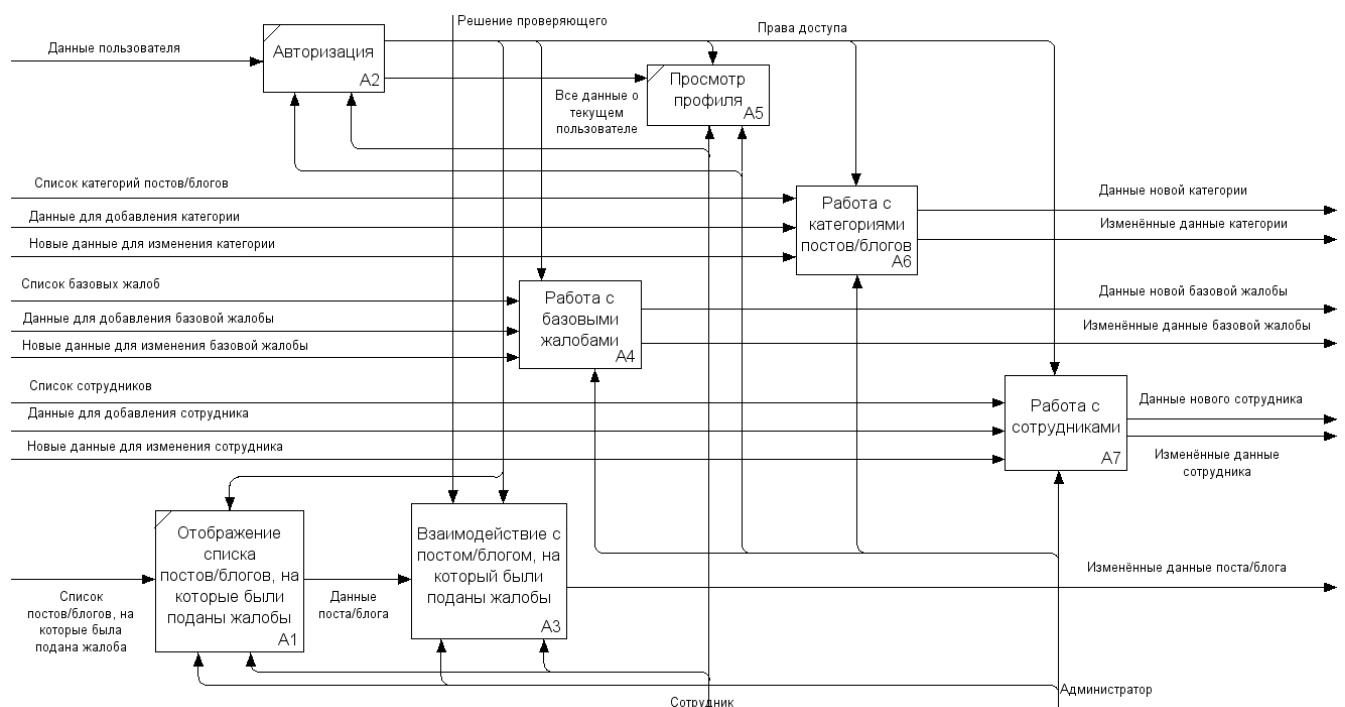


Рисунок 20 – Подробная диаграмма IDEF0 (для сотрудника)

Подробная диаграмма «Взаимодействие с выбранным постом/блогом, на который были поданы жалобы», которая показана на рисунке 21, раскрывает функционирование этого блока. Пользователь может просмотреть все жалобы на пост/блог и потом заблокировать пост или отменить жалобу.

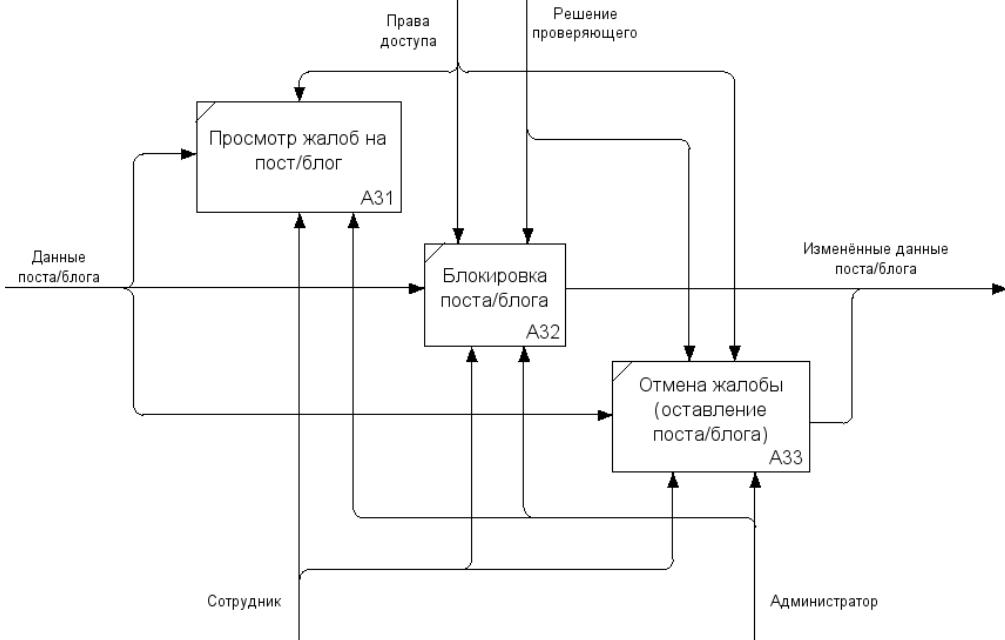


Рисунок 21 - Подробная диаграмма IDEF0 (Взаимодействие с выбранным постом/блогом, на который были поданы жалобы)

Функционирование блока «Работа с базовыми жалобами» раскрыто на рисунке 22. Пользователь может добавить базовую жалобу или посмотреть все с возможностью изменения или удаления выбранной из списка.

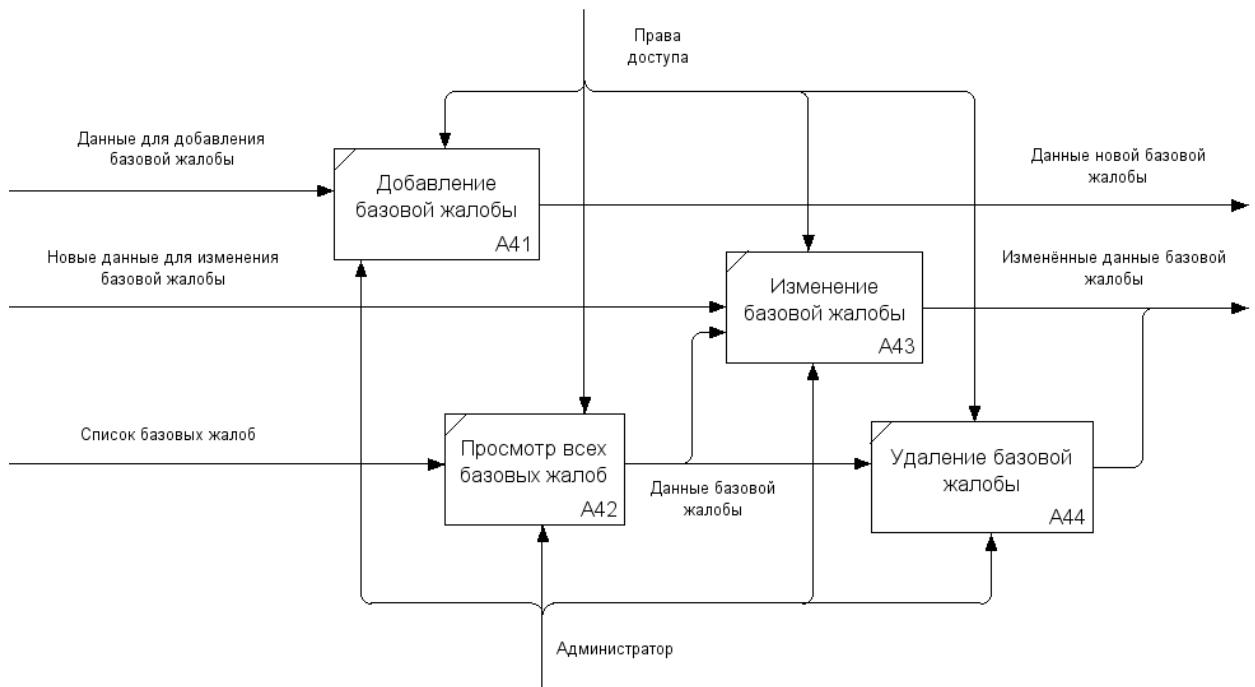


Рисунок 22 - Подробная диаграмма IDEF0 (Работа с базовыми жалобами)

Изм.	Лист	№ докум.	Подпись	Дата

Рисунок 23 показывает функционирование блока «Работа с категориями постов/блогов», а рисунок 24 – «Работа с сотрудниками». На диаграммах, как и в предыдущей, отображается возможность добавления и просмотра всего списка с возможностью изменения или удаления компонентов.

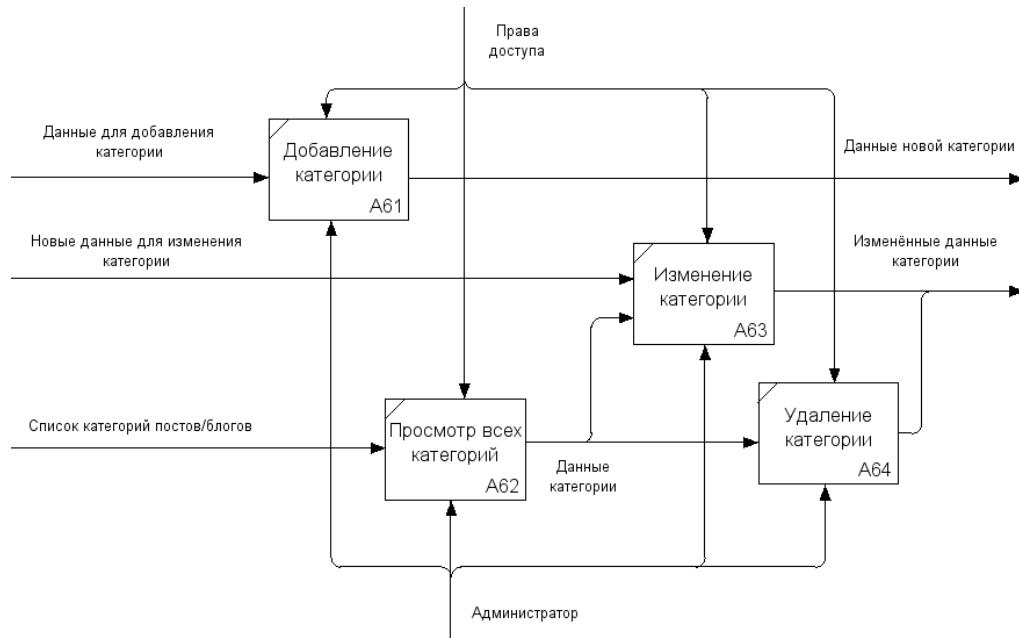


Рисунок 23 - Подробная диаграмма IDEF0 (Работа с категориями постов/блогов)

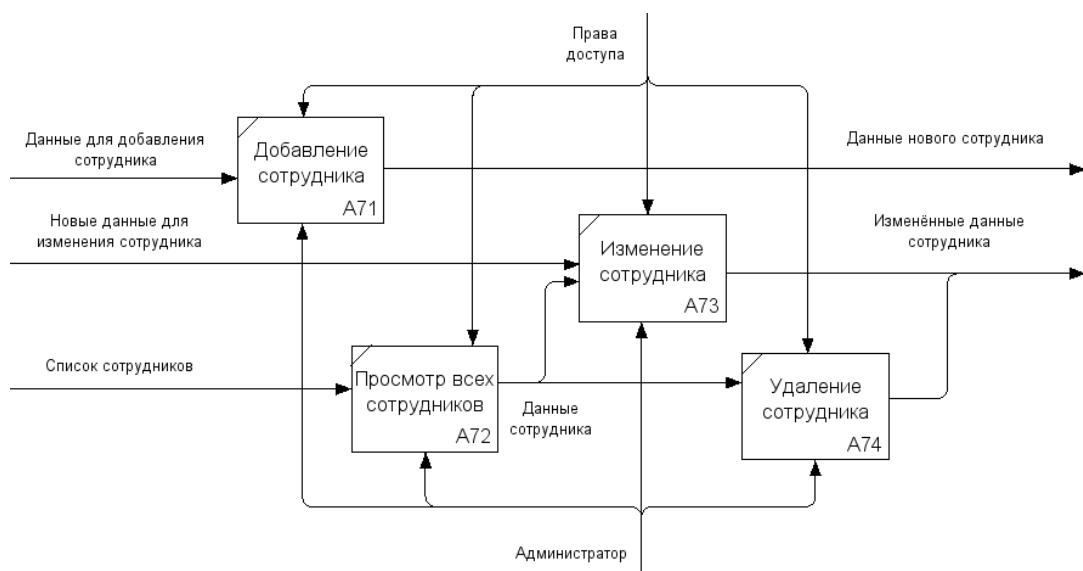


Рисунок 24 - Подробная диаграмма IDEF0 (Работа с сотрудниками)

## 2.3 Диаграммы потоков данных

DFD (Data Flow Diagram) - это методология моделирования бизнес-процессов, которая используется для визуализации потока данных и управления информацией в системе. DFD представляет собой графическое представление процессов, данных и потоков между ними. Она состоит из блоков, которые представляют различные процессы, и стрелок, которые показывают направление потока данных между процессами.

Диаграмма потоков данных, разработанная для приложения «Социальная сеть для фотографов» показана на рисунке 25.

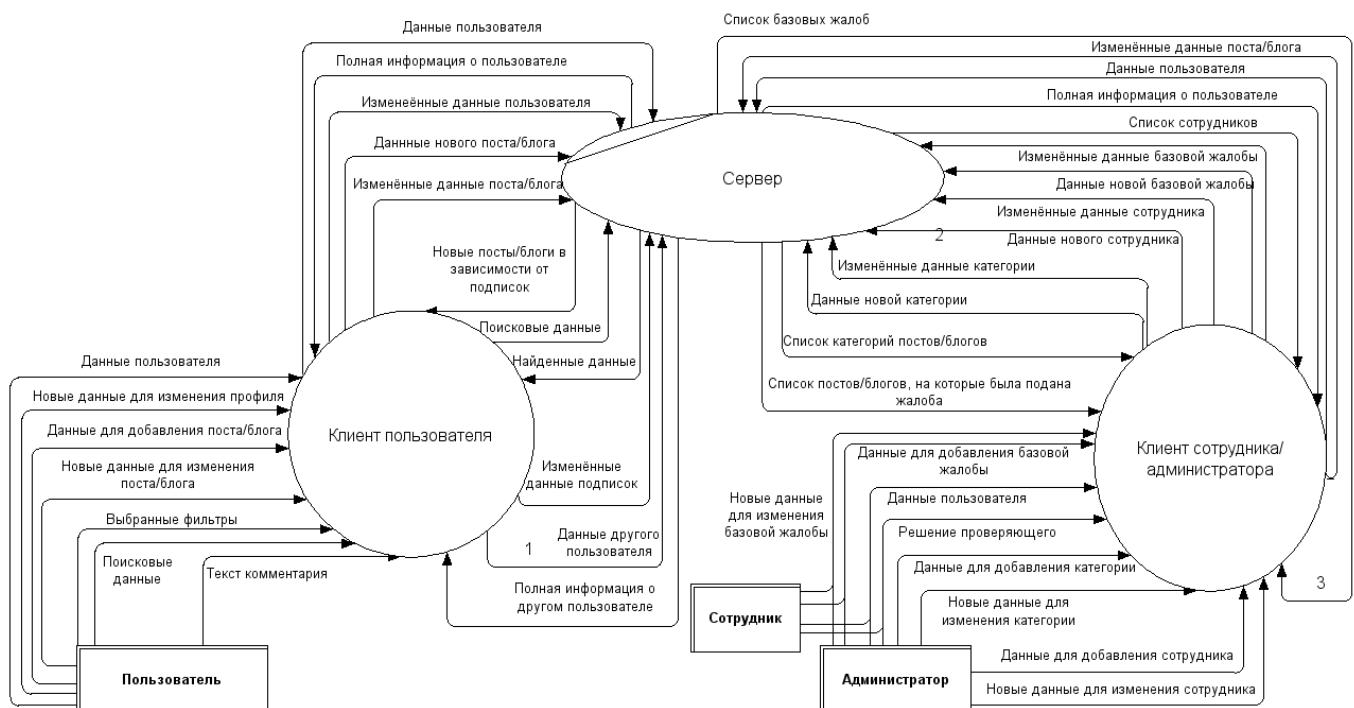


Рисунок 25 - Диаграмма DFD

Приложение состоит из сервера и двух клиентов: для пользователя и для сотрудника/администратора. Следовательно, есть три сущности на диаграмме: пользователь (взаимодействует с клиентом для пользователей), сотрудник и администратор (взаимодействует с клиентом для сотрудника/администратора).

Изм.	Лист	№ докум.	Подпись	Дата

Подробная диаграмма потоков данных, представленная на рисунке 11, описывает детальную работу клиента пользователя.

Пользователь заполняет данные для авторизации, и они отправляются на сервер. При удачной авторизации с сервера приходит полная информация о пользователе после чего пользователь может взаимодействовать с профилем, просматривать ленту или осуществлять поиск.

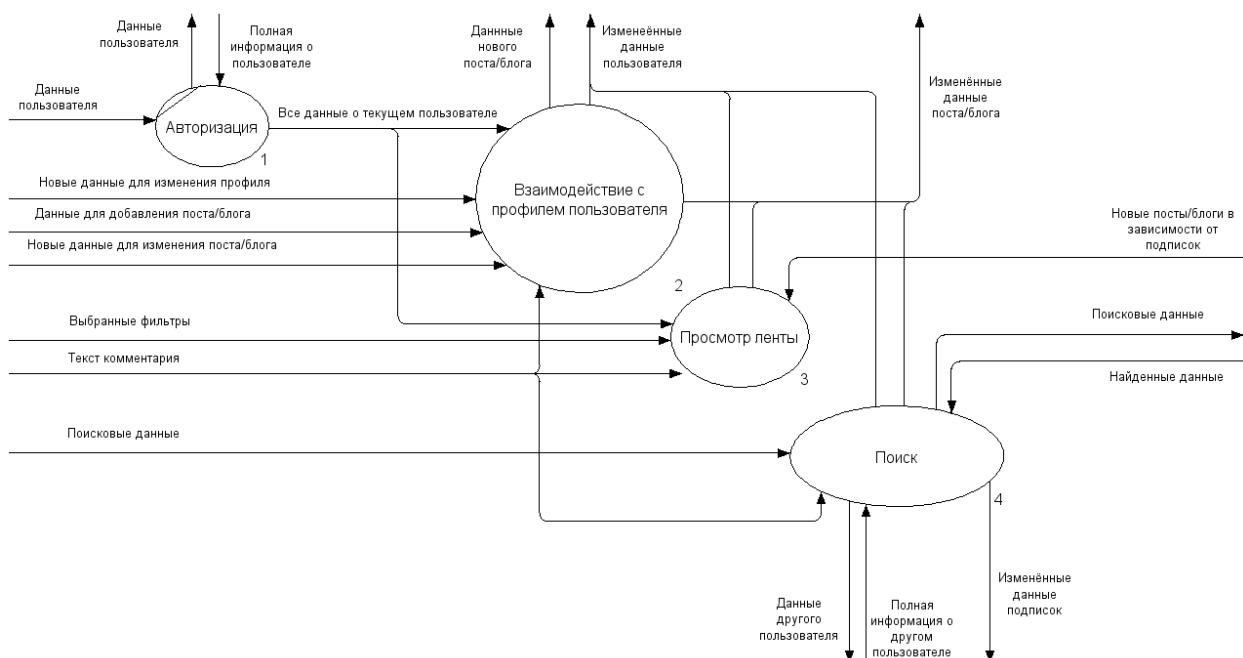


Рисунок 26 - Подробная диаграмма DFD (Клиент пользователя)

Подробная диаграмма потоков данных, представленная на рисунке 27, описывает детальную работу «Взаимодействие с профилем пользователя».

После того как пришли все данные пользователя с сервера они отображаются в профиле, а также отображаются все посты/блоги. Также пользователь может изменить профиль отправив новые данные для изменения профиля и потом эти данные уходят на сервер. При добавлении поста/блога пользователь загружает данные, которые также отправляются на сервер.

Изм.	Лист	№ докум.	Подпись	Дата

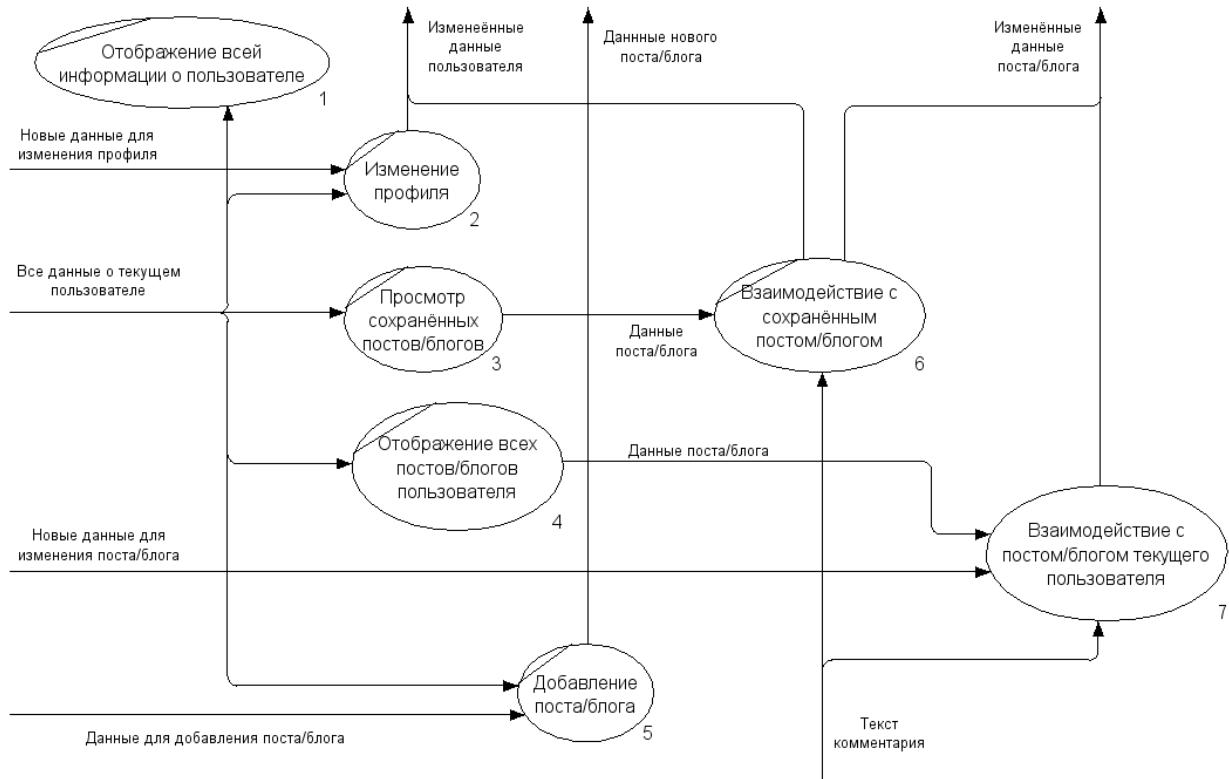


Рисунок 27 - Подробная диаграмма DFD (Взаимодействие с профилем пользователя)

Диаграмма потоков данных «Взаимодействие с постом/блогом текущего пользователя» представленная на рисунке 28. После получение всех данных о посте блоге пользователь может посмотреть все комментарии или добавить свой введя текст, который отправляется на сервер. Также пользователь может удалить пост/блог или поставить/удалить лайк, что приведёт к изменению информации о посте/блоге и изменённые данные отправятся на сервер.

Изм.	Лист	№ докум.	Подпись	Дата

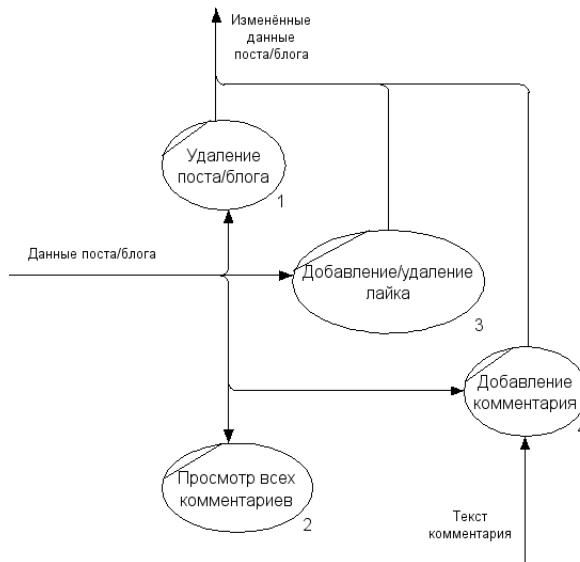


Рисунок 28 - Подробная диаграмма DFD (Взаимодействие с постом/блогом текущего пользователя)

Подробная диаграмма потоков данных, представленная на рисунке 29, описывает детальную работу блока «Поиск». Пользователь вводит поисковые данные, которые отправляются на сервер. После того как с сервера приходят найденные данные они отображаются на сайте, после чего можно поработать с определённым компонентом из списка.

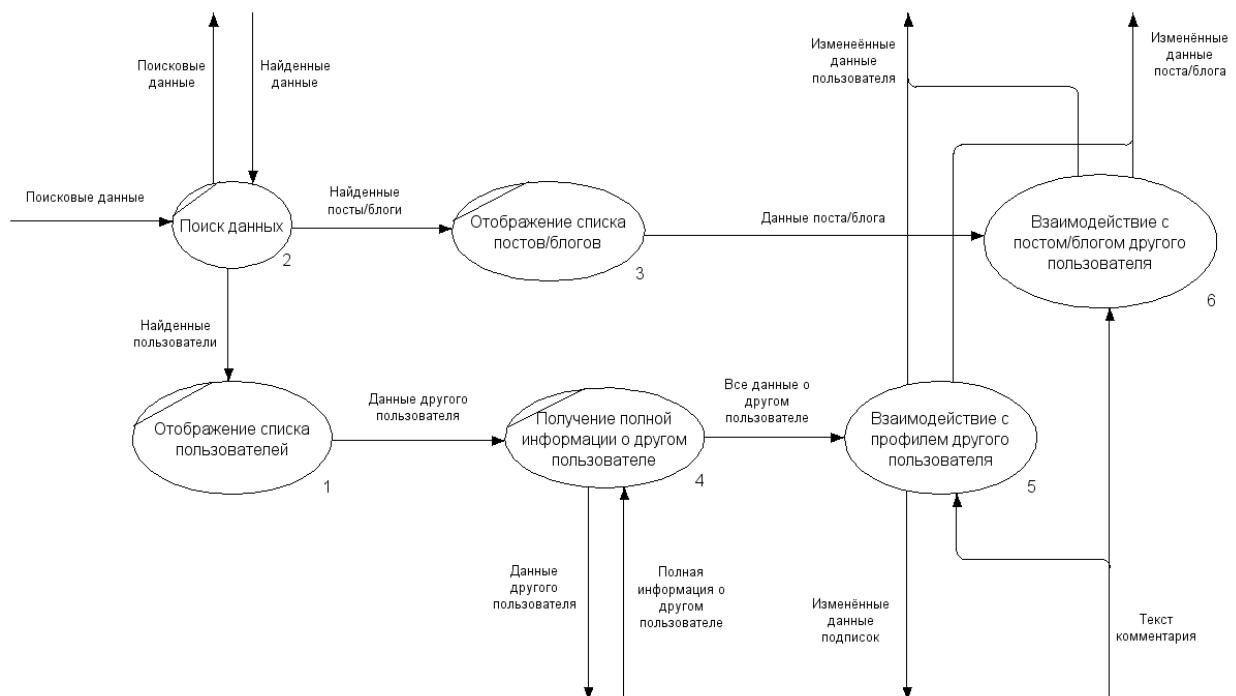


Рисунок 29 - Подробная диаграмма DFD (Поиск)

Изм.	Лист	№ докум.	Подпись	Дата

На рисунке 30 представлена подробная диаграмма потоков данных блока «Взаимодействие с профилем другого пользователя». После получения информации о пользователе с сервера происходит её отображение в профиле. Также можно оформить подписку или отписаться от пользователя и тогда изменённые данные о подписках уходят на сервер.

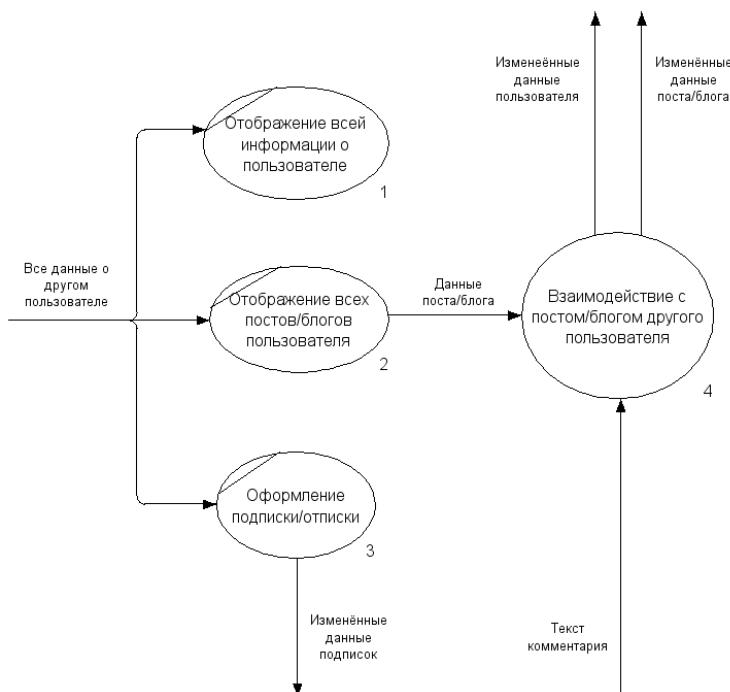


Рисунок 30 - Подробная диаграмма DFD (Взаимодействие с профилем другого пользователя)

Для рассмотрения потоков данных при взаимодействии с постом/блогом другого пользователя была составлена подробная диаграмма на рисунке 31. После получения информации о пользователе с сервера происходит её отображение в профиле. Также можно оформить подписку или отписаться от пользователя и тогда изменённые данные о подписках уходят на сервер.

Изм.	Лист	№ докум.	Подпись	Дата

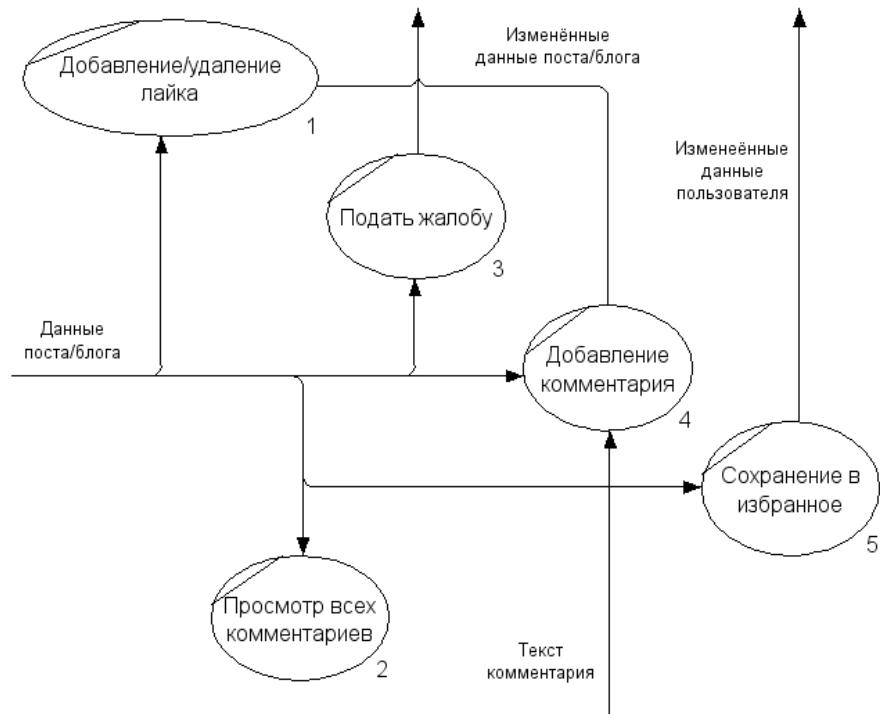


Рисунок 31 - Подробная диаграмма DFD (Взаимодействие с постом/блогом другого пользователя)

На рисунке 32 представлена подробная диаграмма потоков данных блока «Просмотр ленты». С сервера приходят посты/блоги в зависимости от подписок пользователя, которые пользователь может фильтровать по своему желанию.



Рисунок 32 - Подробная диаграмма DFD (Просмотр ленты)

Изм.	Лист	№ докум.	Подпись	Дата

Подробная диаграмма потоков данных для клиента сотрудника/администратора представлена на рисунке 33.

Здесь также пользователь вначале авторизуется, отправляя определённые данные на сервер, затем он может просматривать профиль, работать с категориями, базовыми жалобами и сотрудниками. Также с сервера приходит список постов и блогов, на которые была подана жалоба.

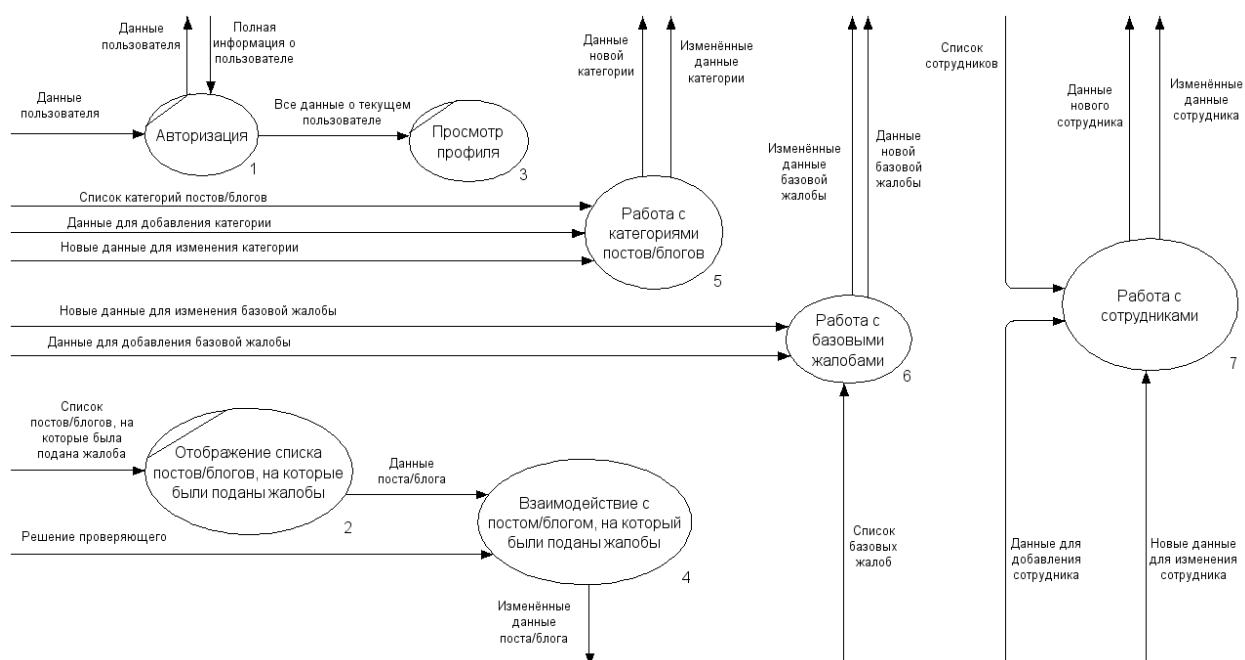


Рисунок 33 - Подробная диаграмма DFD (Клиент сотрудника/администратора)

Для рассмотрения потоков данных при взаимодействии с постом/блогом, на который были поданы жалобы, была составлена подробная диаграмма на рисунке 34. Пользователь может просмотреть все жалобы и заблокировать пост/блог или отменить жалобы. После решения пользователя на сервер отправляются изменённые данные.

Изм.	Лист	№ докум.	Подпись	Дата

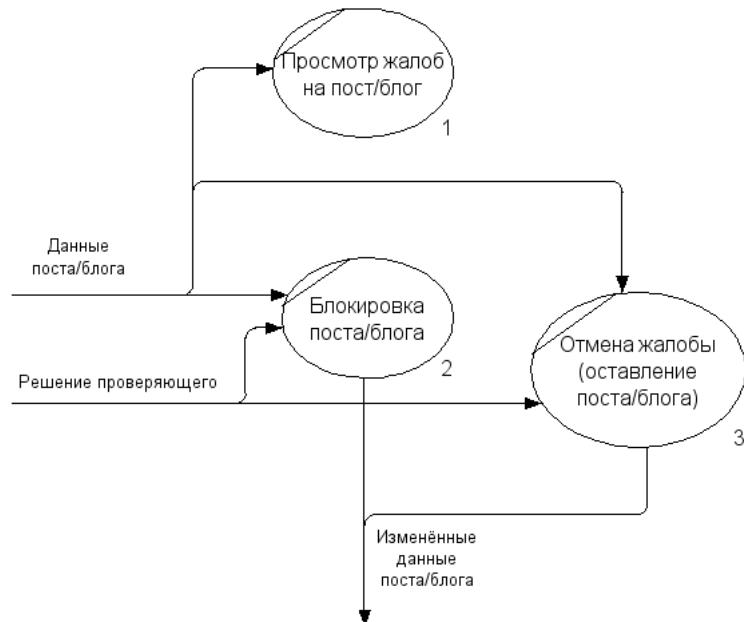


Рисунок 34 - Подробная диаграмма DFD (Взаимодействие с постом/блогом, на который были поданы жалобы)

На рисунках 35, 36, 37 показаны подробные диаграммы по работе с базовыми жалобами, категориями постов/блогов и сотрудниками. Для каждого с сервера приходит полный список. Пользователь может изменить или удалить выбранный элемент из списка, после чего на сервер отправляется изменённая информация. Также пользователь может передать данные для добавления, тогда на сервер отправится новый компонент.

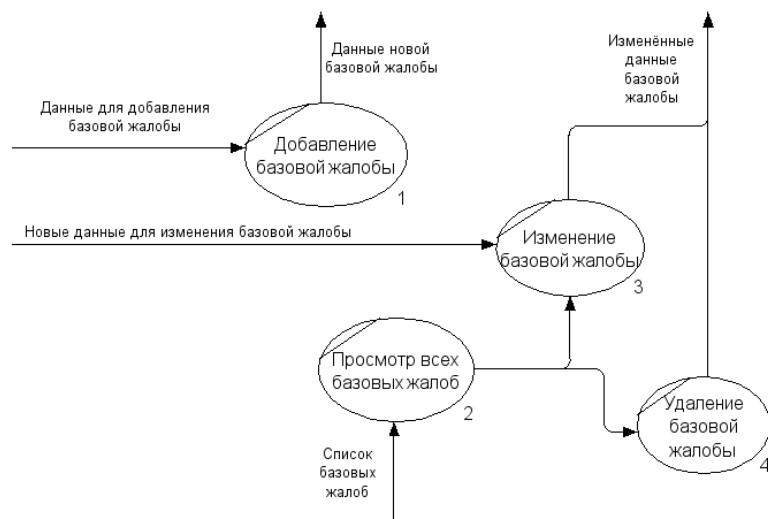


Рисунок 35 - Подробная диаграмма DFD (Работа с базовыми жалобами)

Изм.	Лист	№ докум.	Подпись	Дата

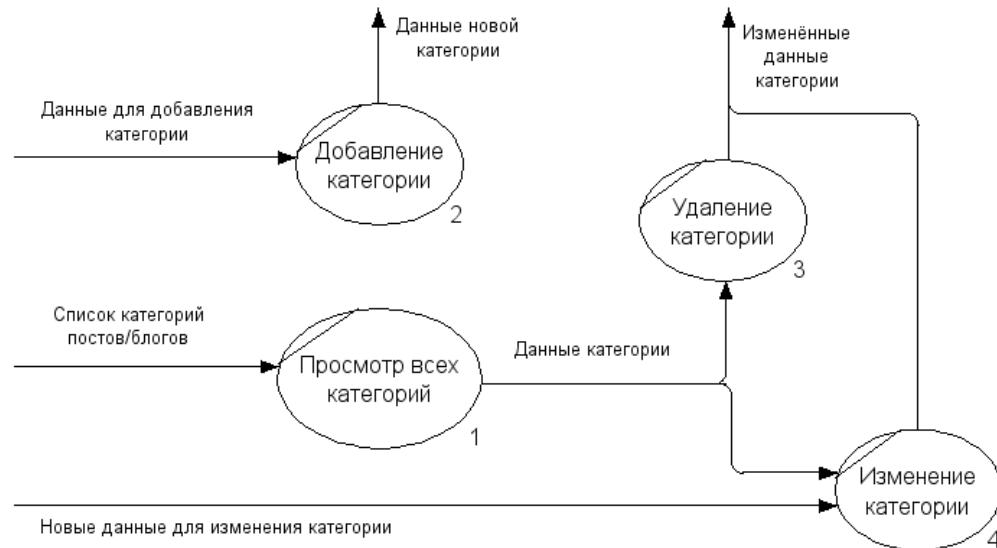


Рисунок 36 - Подробная диаграмма DFD (Работа с категориями постов/блогов)

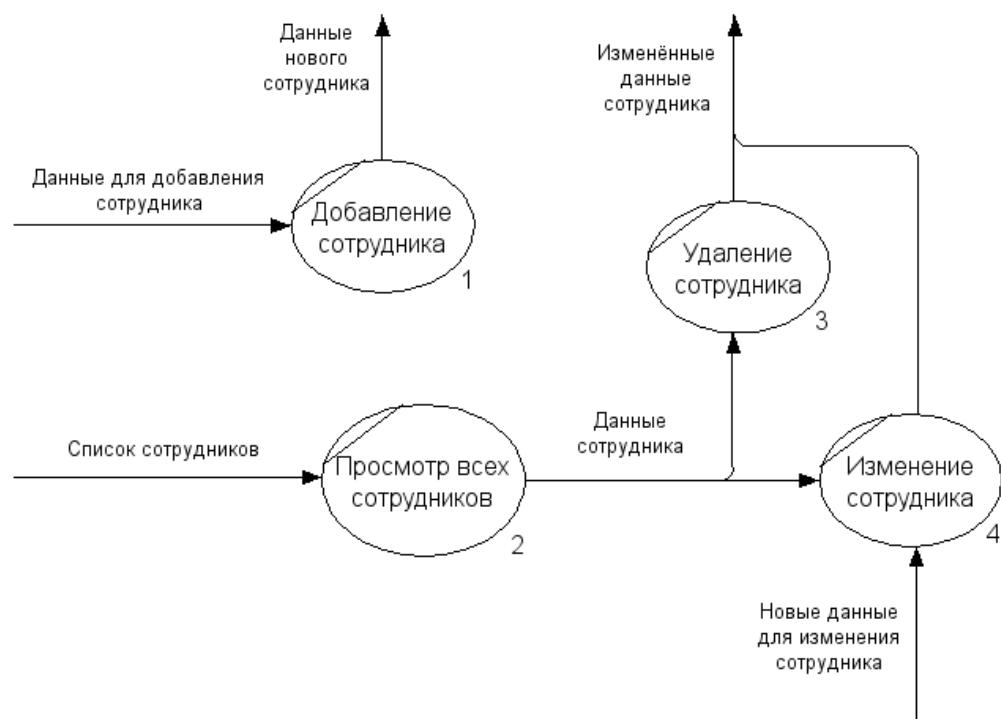


Рисунок 37 - Подробная диаграмма DFD (Работа с сотрудниками)

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

## 2.4 Разработка моделей данных

Разработка моделей данных является важным этапом в процессе проектирования информационной системы. Модель данных определяет структуру и связи между данными, которые будут храниться в БД. Качество модели данных напрямую влияет на эффективность работы БД и возможность ее расширения в будущем. В этом разделе будут рассмотрены концептуальная, логическая и физическая модели данных.

### 2.4.1 Концептуальная модель данных

Концептуальная модель — это отражение предметной области, для которой разрабатывается база данных. Так, все сущности обозначаются в виде прямоугольника. Атрибуты, характеризующие объект - в виде овала, а связи между объектами - ромбами. Мощность связи обозначаются стрелками (в направлении, где мощность равна многим - двойная стрелка, а со стороны, где она равна единице - одинарная).

Концептуальная модель, разработанная для БД Сети для фотографов, представлена на рисунке 38. Объектами на разработанной модели являются Фотограф, Подписчик, Пост/блог, Фотография, Лайк, Избранное, Комментарий, Категория, Папка для категорий. Подписчик может подписаться на Фотографа. Фотограф может создать Пост/блог, который включает Фотографии, Лайки и Комментарии. Лайки ставит фотограф. Комментарии пишет фотограф. Пост/блог принадлежит к определённой категории, которая включается в определённую папку. Также фотограф добавляет в избранное Пост/блог. Пост/блог может содержать Жалобы.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

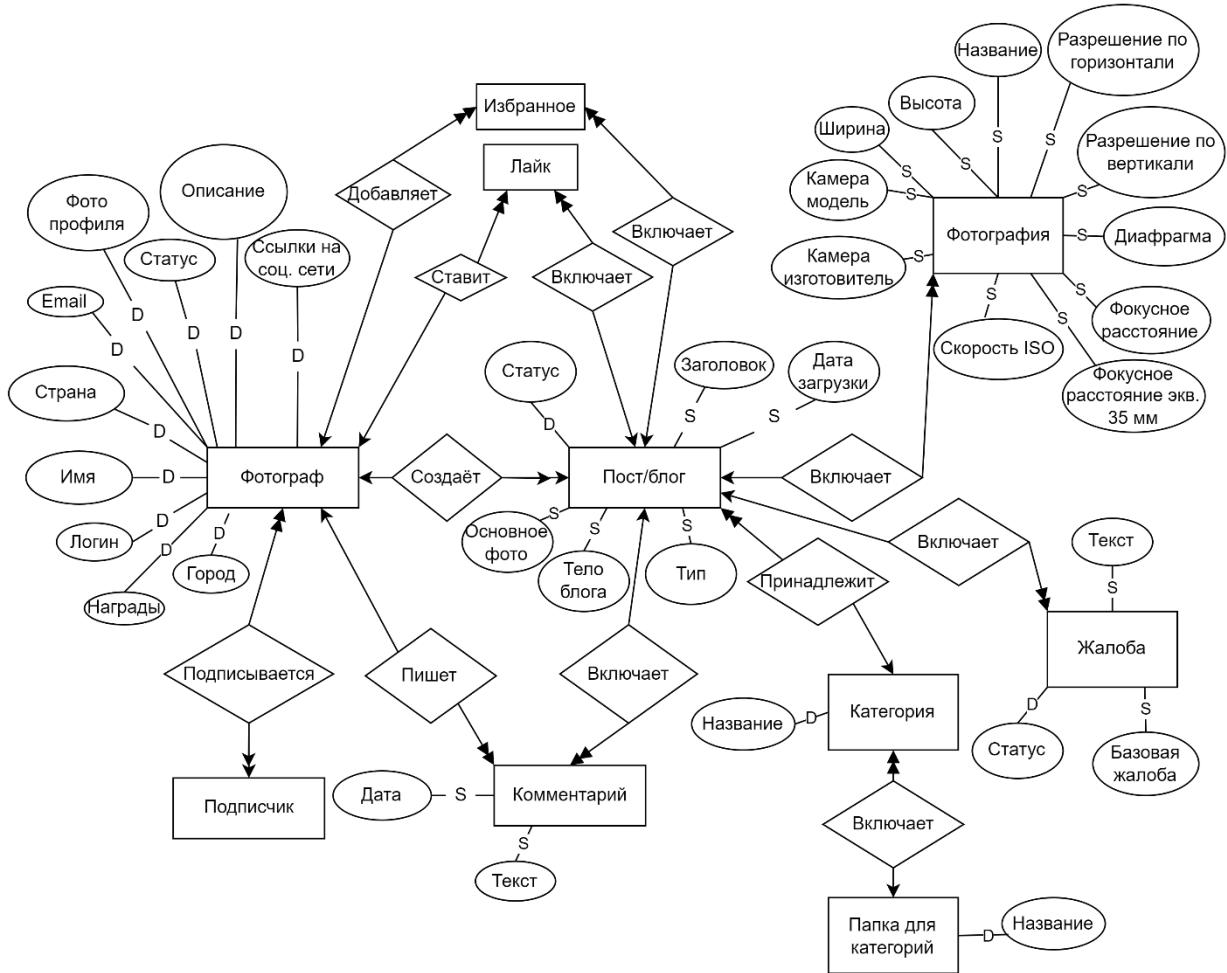


Рисунок 38 - Концептуальная модель данных

#### 2.4.2 Логическая модель данных

Целью построения логической модели является получение графического представления логической структуры исследуемой предметной области. Логическая модель предметной области иллюстрирует сущности, а также их взаимоотношения между собой.

Логическая модель должна читаться по схеме: <Сущность 1> — <отношение / влияние> — <Сущность 2>. Чтение логической модели, представленной на рисунке 39: Фотограф содержит Информацию о фотографе и Подписки. Фотограф создаёт пост/блог, который включает фотографии, лайки, комментарии и жалобы. Фотограф ставит лайки и пишет комментарии, а также добавляет посты/блоги в избранное. Пост/блог принадлежит

Изм.	Лист	№ докум.	Подпись	Дата

категориям, которые включены в папки для категорий. Фотография содержит информацию о фотографии, а жалоба принадлежит базовой жалобе.

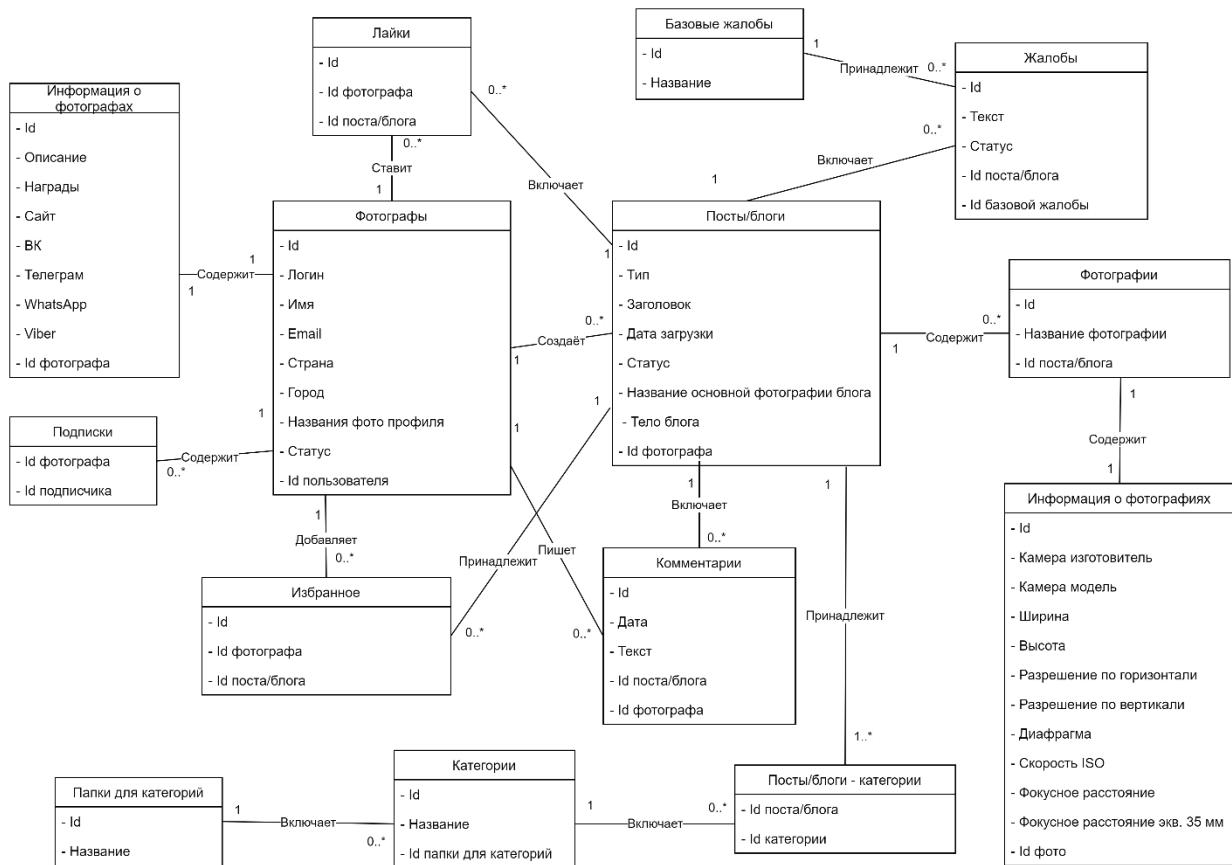


Рисунок 39 - Логическая модель данных

#### 2.4.3 Физическая модель данных

Физическая модель данных – это тип модели данных, который определяет, как данные будут храниться и организовываться в базе данных на уровне конкретных физических объектов.

Физическая модель является продолжением логической модели данных и представляет собой уже конкретное описание базы данных, готовое к реализации на конкретной СУБД. Физическая модель данных включает в себя определение структуры таблиц, атрибутов и их типов, определение связей между таблицами.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

Физическая модель, разработанная для БД приложения «Социальная сеть для фотографов» представлена на рисунке 40.

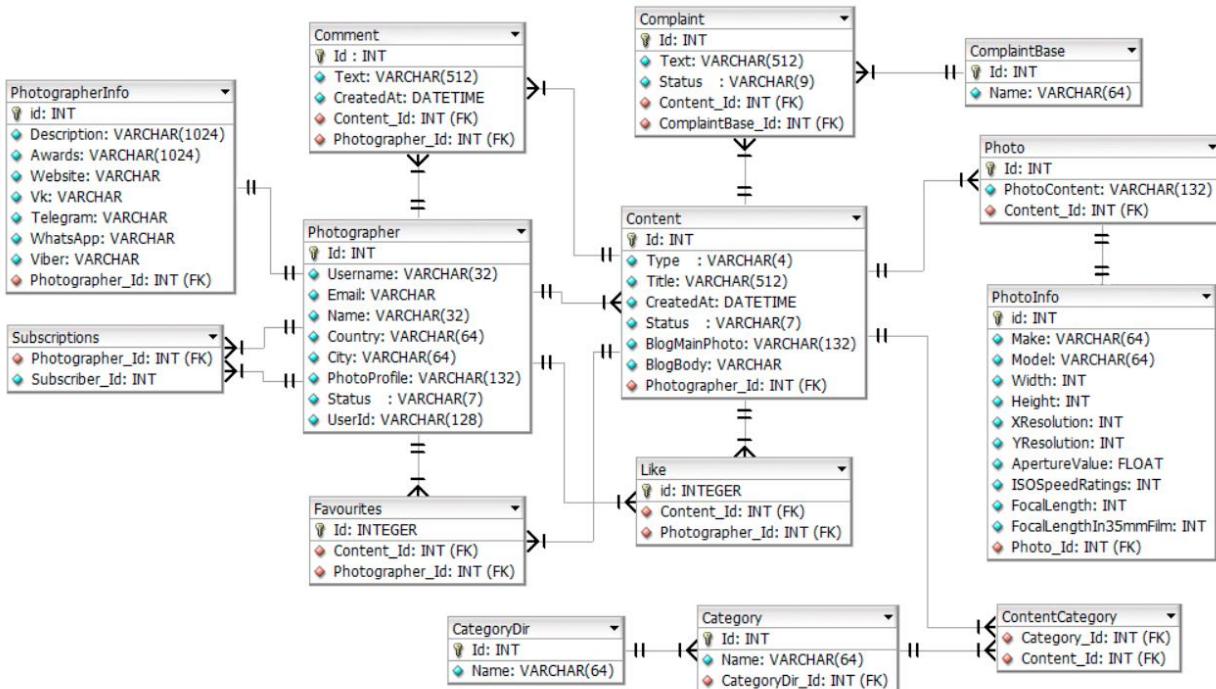


Рисунок 40 - Физическая модель данных

Изм.	Лист	№ докум.	Подпись	Дата

### 3 Реализация программного продукта

#### 3.1 Выбор и обоснование алгоритмов

##### 3.1.1 EntityFramework

EntityFramework (EF) — это мощный инструмент для работы с базами данных, который предоставляет разработчикам множество инструментов и функциональности для упрощения и ускорения процесса разработки приложений. Одним из ключевых компонентов EF является DbContext.

DbContext представляет собой объект, который предоставляет доступ к базе данных и управляет жизненным циклом объектов данных. Он является прослойкой между приложением и базой данных и позволяет разработчикам работать с данными в виде объектов и классов, а не с SQL-запросами и таблицами. DbContext предоставляет разработчикам множество инструментов и функциональности для работы с данными, таких как механизмы отслеживания изменений, механизмы кэширования данных, поддержка транзакций и т.д. DbContext также позволяет использовать LINQ для написания запросов к базе данных, что упрощает работу с данными. [5]

##### 3.1.2 Разделение логики на сервере

Разделение сервера на слои контроллеров (controllers) и сервисов (services) является распространенной практикой в разработке приложений, особенно веб-приложений. Это помогает улучшить архитектуру приложения и сделать его более гибким и масштабируемым.

Контроллеры обычно отвечают за обработку входящих запросов от клиента и возвращение ответов.

Сервисы же отвечают за бизнес-логику и обработку данных. Они предназначены для выполнения операций с данными, а не для получения их из запросов. Сервисы могут взаимодействовать с базой данных, файловой

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

системой, сторонними сервисами и т.д. Они могут содержать сложную бизнес-логику, множество операций и обработчиков ошибок.

Разделение сервера на контроллеры и сервисы помогает создать более гибкую и масштабируемую архитектуру. Оно позволяет легко изменять логику обработки запросов и бизнес-логику отдельно друг от друга.

### 3.1.3 Авторизация и регистрация

Авторизация и регистрация были разработаны с помощью библиотеки аутентификации Microsoft Identity. Microsoft Identity предоставляет разработчикам инструменты для авторизации и регистрации пользователей в приложениях и сервисах. Она позволяет создавать безопасные механизмы аутентификации и авторизации, что обеспечивает защиту данных и удобство использования приложений. [4]

На клиентах аутентификация происходит с помощью куки — это механизм аутентификации пользователей в веб-приложении, который использует зашифрованные куки для хранения информации об аутентификации на стороне клиента. При каждом запросе на сервер, кука отправляется на сервер для проверки аутентификации пользователя. Этот метод является простым, удобным и безопасным для использования.

### 3.1.4 Запросы без перезагрузки страницы

Для возможности выполнения запросов без перезагрузки страницы были рассмотрены две библиотеки. Axios и jQuery — это две разные библиотеки JavaScript, которые могут быть использованы для выполнения AJAX-запросов без перезагрузки страницы в веб-приложениях.

Одним из преимуществ Axios перед jQuery является то, что Axios является более легковесным и удобным в использовании. Она предоставляет

Изм.	Лист	№ докум.	Подпись	Дата

более современную и удобную синтаксическую структуру, которая позволяет упростить процесс написания кода и улучшить его читаемость.

Некоторые из преимуществ Axios по сравнению с jQuery:

1. Axios поддерживает промисы, что позволяет выполнять асинхронный код более удобным и читаемым способом;
2. Axios имеет более современную и удобную синтаксическую структуру, что упрощает процесс написания кода;
3. Axios имеет более надежный и безопасный механизм обработки ошибок, чем jQuery;
4. Axios имеет более высокую производительность, чем jQuery, так как она является более легковесной и оптимизированной для выполнения AJAX-запросов.

### 3.2 Руководство программиста

#### 3.2.1 Сервер

##### 3.2.1.1 Модели и DataContext

Этот класс `DataContext` является классом контекста базы данных, который наследуется от класса `DbContext` в Entity Framework. Он представляет собой прослойку между базой данных и приложением, которая предоставляет доступ к данным и управляет их жизненным циклом.

В данном классе определены свойства `DbSet`, которые представляют собой коллекции объектов, соответствующих таблицам в базе данных. Каждый `DbSet` соответствует одной таблице в базе данных, и его свойства представляют собой колонки этой таблицы.

Листинг класса `DataContext`:

```
public class DataContext : DbContext
{
    public DbSet<Category> Categories { get; set; }
    public DbSet<CategoryDir> CategoryDirs { get; set; }
    public DbSet<Comment> Comments { get; set; }
    public DbSet<ComplaintBase> ComplaintsBase { get; set; }
```

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

```

    public DbSet<Complaint> Complaints { get; set; }
    public DbSet<Content> Contents { get; set; }
    public DbSet<Favourite> Favourites { get; set; }
    public DbSet<Like> Likes { get; set; }
    public DbSet<Photo> Photos { get; set; }
    public DbSet<Photographer> Photographers { get; set; }
    public DbSet<PhotographerInfo> PhotographersInfo { get; set; }
    public DbSet<PhotoInfo> PhotosInfo { get; set; }
    public DbSet<Subscription> Subscriptions { get; set; }

    public DataContext(DbContextOptions<DataContext> options) : base(options)
    {
    }
}

```

### 3.2.1.2 Контроллеры

Контроллеры в ASP.NET Core играют важную роль в обработке HTTP-запросов и определении того, какие данные и каким образом будут переданы между клиентом и сервером.

Контроллеры могут быть использованы для обработки различных типов запросов, таких как GET, POST, PUT, DELETE и другие. Для этого над методом пишутся соответствующие аннотации: [HttpGet], [HttpPost], [HttpPut], [HttpDelete].

Также в методах контроллера вызываются определённые методы из сервиса и определяется в каком виде придёт ответ пользователю – будь то успешный ответ или ошибка.

Пример метода контроллера:

```

[HttpPut]
public async Task<ActionResult<Photographer>>
UpdatePhotographer(UpdatePhotographerDto photographerDto)
{
    try
    {
        return Ok(await _photographersService.UpdatePhotographer(photographerDto));
    }
    catch (NotFoundException ex)
    {
        return NotFound(new NotFoundResponse(ex.Message));
    }
    catch (UniqueFieldException ex)
    {
        return Conflict(new FieldResponse(ex.Field, ex.Message));
    }
}

```

Изм.	Лист	№ докум.	Подпись	Дата

### 3.2.1.3 Авторизация и регистрация

Рассмотрим метод для авторизации пользователя. Он получает объект LoginDto, содержащий данные пользователя, такие как имя пользователя и пароль. Затем метод ищет пользователя в базе данных по заданному имени пользователя или адресу электронной почты и проверяет, совпадает ли введенный пароль с паролем пользователя в базе данных. Если проверка не проходит, метод выбрасывает исключение.

Если пользователь успешно проходит аутентификацию, метод создает объект TokenDto, который содержит информацию о пользователе и выдает ему токен доступа. Затем метод формирует объект JwtSecurityToken, который содержит информацию о пользователе и подписывает его, используя секретный ключ, указанный в конфигурации. Наконец, метод возвращает объект TokenDto с токеном доступа и дополнительной информацией, такой как идентификатор пользователя и его роль в системе.

Листинг метода авторизации (в сервисе):

```
public async Task<TokenDto> Login(LoginDto loginDto)
{
    var user = await FindAppUserByUsernameOrEmail(loginDto.Login);
    if (user == null || !(await _userManager.CheckPasswordAsync(user,
        loginDto.Password)))
    {
        throw new NotFoundException(nameof(AppUser), null);
    }

    var tokenDto = new TokenDto();
    var authClaims = new List<Claim>
    {
        new Claim(ClaimTypes.Name, user.Id),
        new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
    };

    var userRole = (await _userManager.GetRolesAsync(user))[0];
    tokenDto.Role = userRole;

    if (userRole == UserRoles.User)
    {
        var photographer = await
_photographersService.GetSimplePhotographerByUserId(user.Id);
        if (photographer == null)
        {
            throw new NotFoundException(nameof(Photographer), user.Id, "userId");
        }
    }
}
```

Изм.	Лист	№ докум.	Подпись	Дата

```

        }

        if (photographer.Status == StatusPhotographer.Blocked)
        {
            throw new AuthenticationException();
        }

        tokenDto.UserId = photographer.Id.ToString();
    }
else
{
    tokenDto.UserId = user.Id;
}

authClaims.Add(new Claim(ClaimTypes.Role, userRole));

var authSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_configuration["JWT:Secret"]));

var token = new JwtSecurityToken(
    issuer: _configuration["JWT:ValidIssuer"],
    audience: _configuration["JWT:ValidAudience"],
    expires: DateTime.Now.AddMonths(1),
    claims: authClaims,
    signingCredentials: new SigningCredentials(authSigningKey,
    SecurityAlgorithms.HmacSha256)
);

tokenDto.Token = new JwtSecurityTokenHandler().WriteToken(token);

return tokenDto;
}

```

Теперь рассмотрим метод регистрации Register(). Он принимает объект RegisterDto, содержащий данные нового пользователя. Метод также принимает роль пользователя, которая определяет его права доступа в системе.

Перед регистрацией нового пользователя метод проверяет, существует ли уже пользователь с таким же логином или email. Затем метод создает новый объект AppUser на основе переданных данных и вызывает метод \_userManager.CreateAsync() для создания записи пользователя в базе данных. Если создание пользователя прошло успешно, метод добавляет пользователю указанную роль, используя метод \_userManager.AddToRoleAsync().

Листинг метода регистрации (в сервисе):

```

private async Task<GetAppUserDto> Register(RegisterDto registerDto, string role)
{
    if (await CheckExistenceUsername(registerDto.Username))
    {

```

Изм.	Лист	№ докум.	Подпись	Дата

```

        throw new UniqueFieldException(nameof(registerDto.Username),
registerDto.Username);
    }

    if (await CheckExistenceEmail(registerDto.Email))
    {
        throw new UniqueFieldException(nameof(registerDto.Email),
registerDto.Email);
    }

    var user = new AppUser(registerDto);

    var result = await _userManager.CreateAsync(user, registerDto.Password);
    if (!result.Succeeded) throw new Exception(result.ToString());

    await _userManager.AddToRoleAsync(user, role);

    return user.ToGetAppUserDto(role);
}

```

### 3.2.1.4 Ограничение прав доступа

ASP.NET Core предоставляет несколько атрибутов, которые позволяют ограничивать доступ к методам контроллера в зависимости от прав доступа пользователя. Рассмотрим атрибуты, которые используются в приложении.

1. [Authorize] – указывает, что для доступа к методу необходима аутентификация.

2. [Authorize(Roles = "User")] – указывает, что для доступа к методу необходима аутентификация и наличие роли "User".

3. [Authorize(Roles = "Admin")] – указывает, что для доступа к методу необходима аутентификация и наличие роли "Admin".

4. [Authorize(Roles = "Admin,Employee")] – указывает, что для доступа к методу необходима аутентификация и наличие ролей "Admin" или "Employee". Таким образом, пользователи с любой из этих ролей смогут получить доступ к методу.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

### 3.2.1.5 CRUD операции

Полноценные CRUD операции разработаны почти для всех моделей данных. Рассмотрим CRUD операции на самой интересной модели – Photographer.

Для создания фотографа используется функция CreatePhotographer. Она создает новый объект Photographer в базе данных на основе переданных данных из объекта CreatePhotographerDto. Она использует транзакцию базы данных и добавляет нового фотографа в DbSet Photographers контекста базы данных, а затем создает новый объект PhotographerInfo на основе идентификатора нового фотографа и добавляет его в DbSet PhotographersInfo контекста базы данных. Если операции выполнились успешно, функция возвращает созданный объект Photographer.

Листинг метода CreatePhotographer (в сервисе):

```
public async Task<Photographer> CreatePhotographer(CreatePhotographerDto
photographerDto)
{
    var photographer = new Photographer(photographerDto);

    using (var transaction = _context.Database.BeginTransaction())
    {
        await _context.Photographers.AddAsync(photographer);
        await _context.SaveChangesAsync();

        var photographerInfo = new PhotographerInfo(photographer.Id);
        await _context.PhotographersInfo.AddAsync(photographerInfo);
        await _context.SaveChangesAsync();

        transaction.Commit();
    }

    return photographer;
}
```

Метода GetPhotographerById используется для поиска объекта Photographer по его идентификатору в базе данных. Если объект найден и не заблокирован, метод возвращает его. Если объект не найден или заблокирован, метод возвращает null или генерирует исключение.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

### Листинг метода GetPhotographerById (в сервисе):

```
public async Task<Photographer?> GetPhotographerById(int id)
{
    var photographer = await _context.Photographers.FindAsync(id);

    if (photographer == null) return null;
    if (photographer.Status == StatusPhotographer.Blocked)
    {
        throw new AuthenticationException();
    }

    return photographer;
}
```

Метод `UpdatePhotographer` используется для обновления данных фотографа и связанного с ним пользователя в базе данных. Если фотограф или пользователь не найдены, метод генерирует исключение `NotFoundException`. Если имя пользователя или адрес электронной почты уже существуют в базе данных, метод генерирует исключение `UniqueFieldException`. Если обновление проходит успешно, метод возвращает обновленный объект `Photographer`.

### Листинг метода UpdatePhotographer (в сервисе):

```
public async Task<Photographer> UpdatePhotographer(UpdatePhotographerDto
photographerDto)
{
    var photographer = (await GetSimplePhotographerById(photographerDto.Id)) ??
        throw new NotFoundException(nameof(Photographer), photographerDto.Id);

    var user = (await GetUserById(photographer.UserId)) ??
        throw new NotFoundException(nameof(AppUser), photographer.UserId);

    if (await CheckExistenceUsername(photographerDto.Username, photographer.UserId))
    {
        throw new UniqueFieldException(nameof(photographerDto.Username),
photographerDto.Username);
    }

    if (await CheckExistenceEmail(photographerDto.Email, photographer.UserId))
    {
        throw new UniqueFieldException(nameof(photographerDto.Email),
photographerDto.Email);
    }

    // Update user
    user.UserName = photographerDto.Username;
    user.Email = photographerDto.Email;

    await _userManager.UpdateAsync(user);

    // Update photographer
    photographer.Update(photographerDto);
```

Изм.	Лист	№ докум.	Подпись	Дата

```

        _context.Entry(photographer).State = EntityState.Modified;
        await _context.SaveChangesAsync();

    return photographer;
}

```

Для удаления фотографа используется метод DeletePhotographer. Помимо фотографа происходит удаление связанного с ним пользователя из базы данных, а также всех связанных с ним объектов и действий. Если фотограф или пользователь не найдены, метод генерирует исключение NotFoundException.

Листинг метода (в сервисе):

```

public async Task DeletePhotographer(int id)
{
    var photographer = (await GetSimplePhotographerById(id)) ??
        throw new NotFoundException(nameof(Photographer), id);

    var user = (await GetUserById(photographer.UserId)) ??
        throw new NotFoundException(nameof(AppUser), photographer.UserId);

    // DeleteUser
    await _userManager.DeleteAsync(user);

    // DeletePhotographer
    photographer.DeleteProfilePhoto();
    await DeletePhotographerActions(id);

    _context.Photographers.Remove(photographer);
    await _context.SaveChangesAsync();
}

```

### 3.2.1.6 Получение списков постов/блогов

Рассмотрим получение списка на примере постов/блогов определённого фотографа. Но важной особенностью получения списков является то, что они приходят по частям, а не все найденные посты/блоги сразу.

Метод GetPhotographerContents возвращает список контента, который был создан определенным фотографом (photographerId) и имеет указанный тип (typeContent). Возвращаемый список содержит данные в формате GetContentForListDto.

Изм.	Лист	№ докум.	Подпись	Дата

Операция получения списка контента, созданного фотографом, начинается с подключения к контексту базы данных `_context`. Затем выполняется запрос к таблице контента в базе данных с помощью метода `Include`, который указывает на то, какие связанные сущности должны быть включены в запрос. В данном случае, метод `Include` был использован для включения связанных сущностей `Photographer` и `Categories`.

Далее, с помощью метода `Where` выполняется фильтрация контента по заданным параметрам: фотографу (`photographerId`), статусу контента (`StatusContent.Open`) и типу контента (`typeContent`).

Далее, контент сортируется по дате создания (`CreatedAt`) в порядке убывания и пропускаются первые элементы в количестве (`part - 1`) \* `_countInPart`, где `_countInPart` – количество элементов в одной части списка, а `part` – номер запрашиваемой части. Наконец, с помощью метода `Take()` из оставшихся элементов выбирается количество элементов равное `_countInPart`.

Затем, полученный список контента конвертируется в список `GetContentForListDto` с помощью метода `ConvertListContents`, который возвращается в качестве результата метода.

Листинг метода `GetPhotographerContents` (в сервисе):

```
public async Task<List<GetContentForListDto>> GetPhotographerContents(int
photographerId, string typeContent, string userId, int part)
{
    var contents = await _context.Contents
        .Include(item => item.Photographer)
        .Include(item => item.Categories)
        .Where(item => item.PhotographerId == photographerId &&
            item.Status == StatusContent.Open &&
            item.Type == typeContent)
        .OrderByDescending(item => item.CreatedAt)
        .Skip((part - 1) * _countInPart).Take(_countInPart)
        .ToListAsync();

    return await ConvertListContents(contents, userId);
}
```

Изм.	Лист	№ докум.	Подпись	Дата

### 3.2.1.7 Поиск

В программе реализовано два типа поиска: фотографов и постов/блогов. Поиск постов/блогов осуществляется по полю Title, а поиск фотографов по полям Username и Name.

Листинг метода поиска фотографов (в сервисе):

```
public async Task<List<GetPhotographerForListDto>> SearchPhotographers(SearchDto searchDto, int part)
{
    var photographers = await _context.Photographers
        .Where(item => item.Status == StatusPhotographer.Open &&
            (EF.Functions.Like(item.Username, $"%{searchDto.SearchData}%") ||
            EF.Functions.Like(item.Name, $"%{searchDto.SearchData}%")))
        .OrderBy(item => item.Id)
        .Skip((part - 1) * _countInPart).Take(_countInPart)
        .ToListAsync();

    return Photographer.ToListGetPhotographerForListDto(photographers);
}
```

Листинг метода поиска постов/блогов (в сервисе):

```
public async Task<List<GetContentForListDto>> SearchContents(SearchDto searchDto, string typeContent, string userId, int part)
{
    var contents = await _context.Contents
        .Include(item => item.Photographer)
        .Include(item => item.Categories)
        .Where(item => item.Type == typeContent &&
            item.Status == StatusContent.Open &&
            EF.Functions.Like(item.Title, $"%{searchDto.SearchData}%"))
        .OrderByDescending(item => item.CreatedAt)
        .Skip((part - 1) * _countInPart).Take(_countInPart)
        .ToListAsync();

    return await ConvertListContents(contents, userId);
}
```

### 3.2.1.8 Новости и фильтрация

Метод GetNews возвращает список постов/блогов, который соответствует заданным параметрам новостей (newsDto) и подпискам фотографа. Возвращаемый список содержит данные в формате GetContentForListDto.

Изм.	Лист	№ докум.	Подпись	Дата

Операция получения списка новостей начинается с извлечения фотографа из базы данных с помощью метода GetPhotographerById. Если фотограф не найден, генерируется исключение NotFoundException.

Затем, с помощью метода GetPhotographerSubscriptionsIds, получаются идентификаторы всех подписок на фотографа.

Далее, выполняется запрос к таблице контента в базе данных, включаем связанные сущности Photographer и Categories. Затем с помощью метода Where выполняется фильтрация контента по заданным параметрам: статусу контента (StatusContent.Open), подпискам на фотографа (subscriptions), типу контента (newsDto.TypeContent) и категориям (newsDto.CategoriesIds).

Далее, контент сортируется по дате создания и определяется часть, которая вернётся пользователю.

Листинг метода GetNews (на сервисе):

```
public async Task<List<GetContentForListDto>> GetNews(NewsDto newsDto, string
userId, int part)
{
    var photographer = (await GetPhotographerById(newsDto.PhotographerId)) ??
        throw new NotFoundException(nameof(Photographer), newsDto.PhotographerId);

    var subscriptions = await GetPhotographerSubscriptionsIds(photographer.Id);

    var contents = await _context.Contents
        .Include(item => item.Photographer)
        .Include(item => item.Categories)
        .Where(item => item.Status == StatusContent.Open &&
            subscriptions.Contains(item.Photographer.Id) &&
            (newsDto.TypeContent == null ? true : item.Type ==
newsDto.TypeContent) &&
            (newsDto.CategoriesIds == null ? true : item.Categories.Any(category
=> newsDto.CategoriesIds.Contains(category.Id))))
        .OrderByDescending(item => item.CreatedAt)
        .Skip((part - 1) * _countInPart).Take(_countInPart)
        .ToListAsync();

    return await ConvertListContents(contents, userId);
}
```

Изм.	Лист	№ докум.	Подпись	Дата

### 3.2.1.9 Получение фотографии

Рассмотрим метод получения фотографии на примере GetPhotoByContentIdAndName. Он позволяет получить фотографию по заданному идентификатору контента (contentId) и имени файла (name).

Операция получения фотографии начинается с вызова метода GetContentPhotoPath из класса FileInteraction, который возвращает путь к файлу фотографии на сервере на основе заданного идентификатора контента и имени файла.

Затем, с помощью метода ReadAllBytesAsync из класса System.IO.File, считывается содержимое файла фотографии в байтовый массив.

Наконец, байтовый массив содержимого файла возвращается в качестве результата метода с указанием типа файла "image/jpeg".

Листинг метода:

```
[HttpGet("{contentId}/{name}")]
public async Task<ActionResult> GetPhotoByContentIdAndName(int contentId, string name)
{
    var filePath = FileInteraction.GetContentPhotoPath(contentId, name);
    var bytes = await System.IO.File.ReadAllBytesAsync(filePath);
    return File(bytes, "image/jpeg");
}
```

## 3.2.2 Клиенты

### 3.2.2.1 Взаимодействие с сервером

Общение с сервером происходит с помощью HTTP запросов. Все шаблоны запросов вынесены в отдельный класс ApiRequest.

Метод Get выполняет HTTP GET запрос к указанному URL с заданным токеном авторизации и возвращает полученный ответ в виде объекта HttpResponseMessage. Он представляет общий шаблон для выполнения GET запросов с авторизационным токеном и обработкой ошибок.

Изм.	Лист	№ докум.	Подпись	Дата

### Листинг метода Get:

```
public static async Task<HttpResponseMessage> Get(string url, string token)
{
    var request = new HttpRequestMessage()
    {
        RequestUri = new Uri(url),
        Method = HttpMethod.Get
    };

    request.Headers.Add("Authorization", "Bearer " + token);

    HttpResponseMessage response = await _client.SendAsync(request);
    if (!response.IsSuccessStatusCode) await ProcessException(response);
    if (response.StatusCode == HttpStatusCode.NoContent)
    {
        throw new ApiException(StatusCodes.Status404NotFound,
HttpStatusCode.NotFound.ToString());
    }

    return response;
}
```

Данный метод выполняет HTTP POST запрос к указанному URL с заданным токеном авторизации и объектом, который необходимо отправить в формате JSON. Метод возвращает полученный ответ в виде объекта HttpResponseMessage. Он представляет общий шаблон для выполнения POST запросов с авторизационным токеном и объектом для отправки, сериализованным в формат JSON, и обработкой ошибок.

### Листинг метода Post:

```
public static async Task<HttpResponseMessage> Post(string url, Object objectToSend,
string token)
{
    var request = new HttpRequestMessage()
    {
        RequestUri = new Uri(url),
        Method = HttpMethod.Post
    };
    request.Headers.Add("Authorization", "Bearer " + token);

    request.Content = new StringContent(
        JsonConvert.SerializeObject(objectToSend),
        Encoding.UTF8,
        "application/json"
    );

    HttpResponseMessage response = await _client.SendAsync(request);
    if (!response.IsSuccessStatusCode) await ProcessException(response);

    return response;
}
```

Изм.	Лист	№ докум.	Подпись	Дата

### 3.2.2.2 Авторизация

Метод CreateCookieAuthentication создает аутентификационные куки на основе полученного объекта TokenDto, используя ClaimsIdentity, ClaimsPrincipal и AuthenticationProperties. Куки сохраняют информацию о токене, роли пользователя и идентификаторе фотографа. Они имеет опции, такие как время жизни и возможность обновления.

Листинг метода авторизации:

```
private async Task CreateCookieAuthentication(TokenDto tokenDto)
{
    var identity = new
ClaimsIdentity(CookieAuthenticationDefaults.AuthenticationScheme, ClaimTypes.Name,
ClaimTypes.Role);
    identity.AddClaims(new List<Claim>
    {
        new Claim(ClaimTypes.Name, Guid.NewGuid().ToString()),
        new Claim(ClaimTypes.Role, tokenDto.Role),
        new Claim("Token", tokenDto.Token),
        new Claim("PhotographerId", tokenDto.UserId)
    });
    var principal = new ClaimsPrincipal(identity);

    var authProperties = new AuthenticationProperties
    {
        AllowRefresh = false,
        ExpiresUtc = DateTimeOffset.Now.AddMonths(1),
        IsPersistent = true,
    };

    await HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme,
new ClaimsPrincipal(principal), authProperties);
}
```

### 3.2.2.3 Представления

Для создания красивых и адаптивных представлений веб-интерфейса была выбрана библиотека Bootstrap. Bootstrap предоставляет широкий набор инструментов для создания различных элементов интерфейса и готовых компонентов, таких как модальные окна, карусели, вкладки и т.д. Кроме того, Bootstrap имеет гибкую систему сеток, которая позволяет создавать адаптивные макеты, которые легко адаптируются к различным размерам экранов и устройствам.

Изм.	Лист	№ докум.	Подпись	Дата

### 3.3 Руководство пользователя

#### 3.3.1 Клиент пользователя

##### 3.3.1.1 Авторизация и регистрация

При запуске приложения появляется страница авторизации, представленная на рисунке 41 в приложении А. После введения правильного логина и пароля открывается главная страница приложения.

Также в главном меню можно перейти на страницу регистрации. Страница регистрации, представленная на рисунке 41 в приложении А. При корректном заполнении всех полей пользователь добавляется в системе и происходит переход на страницу авторизации.

##### 3.3.1.2 Главное меню

Главное меню состоит из следующих элементов: «Профиль», «Главная», «Поиск», «Добавить», «Избранное», «Настройки», «Выход». Каждый из элементов отвечает за переход на соответствующую страницу.

##### 3.3.1.3 Профиль

В профиле можно посмотреть информацию о фотографе во вкладке «Подробнее» и все посты/блоги во вкладках «Пост» и «Блог» соответственно (рисунок 42 в приложении А). Также нажав на кнопки около количества подписок и подписчиков, появляются модальные окна с отображением списков фотографов (рисунок 43 в приложении А).

В профиле другого пользователя (рисунки 44 в приложении А) есть кнопка для подписки или отписки на данного фотографа.

Изм.	Лист	№ докум.	Подпись	Дата

### 3.3.1.4 Настройки

В настройках есть возможность изменения профиля, пароля или удаления аккаунта, что продемонстрировано на рисунке 45 в приложении А. Также на данном рисунке показана страница для изменения пароля.

Изменение профиля фотографа показано на рисунке 46 в приложении А. При изменении профиля можно изменить фото профиля, выбрав соответствующий файл и нажав на кнопку «Сохранить». Также во вкладках «Основная инф» и «Доп инф» можно изменить информацию, которая будет отображаться в профиле пользователя.

### 3.3.1.5 Добавление поста/блога

Перейти на страницу с добавлением поста/блога можно из меню приложения, выбрав соответствующие пункты меню.

При добавлении поста (рисунок 47 в приложении А) необходимо обязательно загрузить одно или несколько фото, добавить описание и выбрать категории, к которым будет отнесён пост. При добавлении фотографии появляется окно с добавлением информации о загруженном фото (рисунок 48 в приложении А). При добавлении блога также загружается основное фото, заголовок и категории, а также с помощью специальных компонентов заполняется блог.

### 3.3.1.6 Лайки, комментарии, избранное, жалобы

Посту/блогу можно поставить лайк, нажав соответствующую кнопку, также можно убрать лайк, если он был поставлен ранее. Аналогично идёт добавление поста/блога в избранное. Нажав у поста на кнопку со значком комментариев, открывается страница поста/блога вместе со всеми

Изм.	Лист	№ докум.	Подпись	Дата

комментариями, где пользователь может добавить свой (рисунок 49 в приложении А).

Также на панели под постом/блогом есть значок отправки жалобы. Нажав на него, можно описать то, что не понравилось в данном контенте (рисунок 50 в приложении А).

### 3.3.1.7 Главная страница

На главной странице (рисунок 51 в приложении А) во вкладке «Новости» отображаются все новостные посты/блоги от фотографов, на которых осуществлена подписка. Во вкладке «Другое» отображаются все остальные посты/блоги фотографов.

Все посты/блоги на главной странице можно отфильтровать с помощью специальных фильтров, представленных на рисунке 52 в приложении А. Контент во вкладке «Новости» можно отсортировать по типу и категориям, во вкладке «Другое» также по тому, что сначала показывать: лучшие или новые. Лучшие также можно отсортировать за заданный период, а новые – по количеству лайков.

### 3.3.1.8 Страница поиска

Скриншоты страницы поиска, представлены на рисунке 53 в приложение А. Для поиска есть специальная строка куда вводятся данные. После нажатия на кнопку поиска, будут отображаться данные в зависимости от той вкладки, в которой находится пользователь. Поиск фотографов производится по полям Логин и Имя. Поиск постов/блогов осуществляется по полю описания (заголовок).

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

### 3.3.2 Клиент сотрудника/администратора

#### 3.3.2.1 Авторизация и главное меню

При запуске приложения появляется страница авторизации. После введения правильного логина и пароля открывается страница с количеством жалоб на пользователей.

Главное меню состоит из следующих элементов: Жалобы, Профиль, Выход. Для администратора также появляются компоненты для перехода на страницы работы с базовыми жалобами, категориями и сотрудниками.

#### 3.3.2.2 Профиль

В профиле сотрудника/администратора отображается вся информация о нём, что представлено на рисунке 54 в приложении А. Также есть две кнопки для изменения профиля и изменения пароля. На рисунке 54 в приложении А продемонстрировано как выглядит страница изменения профиля.

#### 3.3.2.3 Жалобы

На основной странице жалоб отображается список с количеством жалоб на фотографов, что продемонстрировано на рисунке 55 в приложении А. При выборе любого фотографа из списка и нажатии на кнопку «Подробнее» пользователь попадает на страницу со списком количества жалоб на посты/блоги фотографа (рисунок 56 в приложении А). Тут он может заблокировать пользователя или перейти к просмотру жалоб на конкретный пост/блог нажав на соответствующую кнопку «Подробнее». При переходе по кнопки «Подробнее» открывается страница с предпросмотром поста/блога и списком всех жалоб на данный контент (рисунок 57 в приложении А). Здесь можно как отклонить каждую из жалоб, так и отклонить все сразу или заблокировать контент.

Изм.	Лист	№ докум.	Подпись	Дата

### 3.3.2.4 Базовые жалобы

Работа с данным компонентом доступна только администратору.

При переходе на соответствующую страницу перед пользователем появляется список всех базовых жалоб (рисунок 58 в приложении А). Также на данном рисунке продемонстрирована страница добавление базовой жалобы. В списке можно выбрать определённую базовую жалобу для изменения или удаления (соответствующие страницы показаны на рисунке 59 в приложении А).

### 3.3.2.5 Категории

Работа с данным компонентом доступна только администратору.

При открытии страницы появляется список папок для категорий и при нажатии на конкретную – список категорий в ней (рисунок 60 в приложении А). Каждый элемент этого списка можно изменить или удалить. Если в папке есть категории или если за категорией закреплены посты/блоги, то удаление невозможно и появляется соответствующая надпись, иначе кнопку удаления будет доступна (рисунок 64 в приложении А). Также есть две страницы для добавления папки и для добавления категории в определённую папку, что продемонстрировано на рисунке 63 в приложении А.

### 3.3.2.6 Сотрудники

Работа с данным компонентом доступна только администратору.

При переходе на соответствующую страницу перед пользователем появляется список всех сотрудников (рисунок 65 в приложении А). При добавлении сотрудника (рисунок 66 в приложении А) пользователь может выбрать его роль, ввести основную информацию и пароль, который сотрудник

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

может потом изменить у себя в профиле. Также можно изменить или удалить любого сотрудника из списка (данные страницы приведены на рисунке 67 в приложении А).

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

МИВУ 09.03.04-8.000 ВКР

Лист

63

## 4 Тестирование программного продукта

Одним из важнейших этапов создания приложения является его тестирований. Целью тестирования является проверка работоспособности программы, правильности выполнения всех функций, а также правильности обработки всех исключений, возникающих в ходе работы программы.

Алгоритмы данной ИС не обладают особой сложностью, и эффективнее будет провести ручное тестирование клиента в связке с сервером для полной проверки работоспособности всех составляющих частей приложения.

В таблице 1 приведён чек-лист для проведения тестирования клиента пользователя.

Таблица 1 - Чек-лист «Тестирование клиента пользователя»

Общие		
№	Проверка	Результат
1	Запуск программы	Для неавторизованного пользователя отображается окно авторизации, иначе главная страница
2	Тестирование главного меню приложения	Для неавторизованного пользователя отображаются два элемента меню: авторизация и регистрация, иначе меню состоит из следующих элементов: профиль, главная страница, поиск, добавление поста/блога, избранное, настройки, выход.  При нажатии на каждый элемент открывается соответствующая страницы, а при нажатии на «Выход» происходит соответственно выход из приложения и переход на страницу авторизации
Авторизация		
№	Проверка	Результат
3	Ввести данные в поля «Логин/Email» и «Пароль», которые существуют в БД, и нажать на кнопку «Войти»	Переход на главную страницу

Продолжение таблицы 1 – Чек-лист «Тестирование клиента пользователя»

№	Проверка	Результат
4	Ввод данных в поля «Логин/Email» и «Пароль», которые не существуют в БД, и нажать на кнопку «Войти»	Появляется ошибка «Неправильный логин или пароль»
5	Не ввести данные в какое-либо из полей и нажать на кнопку «Войти»	Появляется ошибка «Поле обязательное для заполнения» под соответствующим полем
Регистрация		
№	Проверка	Результат
6	Ввести корректные данные в поля для регистрации и нажать на кнопку «Зарегистрироваться»	Переход на страницу авторизации
7	Не ввести данные в какое-либо из полей	Появляется ошибка «Поле обязательное для заполнения» под соответствующим полем
8	Ввести «Логин» или «Пароль» короче 4 или длиннее 32 символов	Появляется ошибка «Должно быть длиннее 4 и короче 32 символов»
9	Ввести некорректный «Email»	Появляются ошибки в соответствии с тем какая из частей email введена неверно
10	Значения в поле «Подтвердите пароль» и «Пароль» не совпадают	Появляется ошибка «Пароли не совпадают»
11	Ввести «Логин» или «Email», который уже используется в системе	Появляется ошибка «Данное значение уже существует» под соответствующим полем
Страница профиля		
№	Проверка	Результат
12	Запуск страницы	<p>Корректно отображается:</p> <ul style="list-style-type: none"> <li>- фотография профиля, логин и имя фотографа</li> <li>- вкладки «Подробнее», «Посты» и «Блоги»</li> </ul> <p>Для профиля другого пользователя отображается кнопка «Подписаться»/«Отписаться»</p>
13	Нажатие на кнопку «Подписаться»/«Отписаться»	<p>Если кнопка «Подписаться»: кнопка изменяется на «Отписаться», и у текущего фотографа добавляется подписчик</p> <p>Если кнопка «Отписаться»: кнопка изменяется на «Подписаться», и у текущего фотографа убавляется подписчик</p>

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 1 – Чек-лист «Тестирование клиента пользователя»

№	Проверка	Результат
14	Отображение вкладки «Подробнее»	<p>Корректно отображается:</p> <ul style="list-style-type: none"> <li>- Количество подписчиков кнопкой для просмотра списка подписчиков</li> <li>- Количество подписок с кнопкой для просмотра списка подписок</li> <li>- Описание (если пользователь добавил)</li> <li>- Страна и город (если пользователь добавил)</li> <li>- Награды (если пользователь добавил)</li> <li>- Ссылка на сайт (если пользователь добавил)</li> <li>- Ссылка на соцсети (если пользователь добавил)</li> </ul>
15	Нажатие на кнопку для просмотра списка подписчиков	Переход на окно, где корректно отображаются все подписчики и их количество
16	Нажатие на кнопку для просмотра списка подписок	Переход на окно, где корректно отображаются все подписки и их количество
17	Отображение вкладки «Посты» и «Блоги»	Отображаются все посты/блоги пользователя отсортированные по дате загрузки
Пост/блог		
№	Проверка	Результат
18	Отображение поста/блога	<p>Корректно отображается:</p> <ul style="list-style-type: none"> <li>- фотография профиля и логин пользователя</li> <li>- дата загрузки</li> <li>- заголовок поста/блога</li> <li>- количество лайков, комментариев и сохранений в избранное с соответствующими кнопками</li> <li>- поставлен лайк или нет текущем пользователем</li> <li>- сохранён в избранное или нет текущем пользователем</li> <li>- кнопка удаления поста, если пост/блог принадлежит текущему пользователю</li> <li>- кнопка для подачи жалобы, если пост/блог принадлежит другому пользователю</li> <li>- значок заблокированного поста/блога при отображении в профиле текущего пользователя</li> </ul> <p>Для поста</p> <ul style="list-style-type: none"> <li>- если фотографий больше 1, то появляется слайдер</li> </ul> <p>Для блога</p> <ul style="list-style-type: none"> <li>- если блог находится в списке, то отображение только главного фото</li> <li>- на странице блога корректно отображается всё содержимое блога</li> </ul> <p>На странице определённого поста/блога отображается список категорий</p>

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 1 – Чек-лист «Тестиирование клиента пользователя»

№	Проверка	Результат
19	Нажатие на кнопку лайка	Если лайк не стоял: меняется картинка на поставленный лайк и добавляется количество лайков  Если лайк стоял: меняется картинка на не поставленный лайк и убавляется количество лайков
20	Нажатие на кнопку избранного	Если пост/блог не был добавлен в избранное: меняется картинка на добавленный в избранное и добавляется количество сохранений в избранное  Если пост/блог был добавлен в избранное: меняется картинка на не добавленный в избранное и убавляется количество сохранений в избранное
21	Нажатие на кнопку комментариев или на надписи «Подробнее...» / «Читать...»	Переход на страницу поста/блога
22	Нажатие на фото или на логин пользователя	Переход на страницу профиля пользователя
23	Нажатие на кнопку удаления поста/блог (только если пост/блог принадлежит текущего пользователя)	Появляется вопрос «Вы уверены что хотите удалить дынний контент?» и возможность выбрать кнопки «Отмена» и «Удалить»  При выборе «Удалить» - пост/блог удаляется, и пользователь переходит в профиль
24	Нажатие на кнопку для подачи жалобы (если пост/блог принадлежит другому пользователю)	Появляется окно с возможностью выбора базовой жалобы и полем куда можно ввести дополнительный текст
Комментарии к посту		
№	Проверка	Результат
25	Открытие страницы поста/блога	Корректно отображается: - количество комментариев - список комментариев
26	Нажатие на фото или логин пользователя у комментария	Переход на страницу профиля пользователя  Если комментарий от текущего пользователя, то отображение страницы профиля текущего пользователя
27	Ввод текста комментария в поле «Написать комментарий...» и нажатие на кнопку отправить	Комментарий отображается в списке комментариев и счётчик комментариев увеличивается

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 1 – Чек-лист «Тестирование клиента пользователя»

Главная страница		
№	Проверка	Результат
28	Запуск страницы	<p>Корректно отображается:</p> <ul style="list-style-type: none"> <li>- вкладка «Новости» со всеми постами от пользователей, на которых подписан текущий пользователь</li> <li>- вкладка «Другое» с предложенными остальными постами</li> </ul>
29	Нажатие на кнопку фильтрации	<p>Если находимся во вкладке «Новости», то фильтрация может осуществляться только по типу контента и категориям</p> <p>Если находимся во вкладке «Другое», то фильтрация также может осуществляться по тому, что показать: новые или лучшие.</p> <p>Если выбирается новые, то появляется выпадающий список с возможностью выбора минимального количества лайков.</p> <p>Если выбирается лучшие, то появляется выпадающий список с возможностью выбора периода.</p>
30	Выбор параметров для фильтрации	Корректно отображаются посты с применённым фильтром

Поиск		
№	Проверка	Результат
31	Ввести данные для поиска и нажать кнопку поиска	<p>Во вкладке «Фотографы» отображаются подходящие под атрибуты поиска пользователи, а во вкладке «Пост» и «Блог» - соответственно посты и блоги</p> <p>Если ничего подходящего нет, то отображается надпись, что ничего не найдено</p>
32	Ввести текст короче 2 символов	Появляется ошибка «Текст должен быть не короче 2 символов»

Добавление поста/блога		
№	Проверка	Результат
33	Корректно добавить все данные	Пост/блог добавляется и осуществляется переход на страницу профиля
34	Не добавить один из компонентов	Появляется ошибка под незаполненным компонентом
35	Попробовать добавить фото, превышающее допустимый лимит	Фотография не добавляется

Продолжение таблицы 1 – Чек-лист «Тестирование клиента пользователя»

Избранное		
№	Проверка	Результат
36	Запуск страницы	Корректно отображается вкладки «Посты» и «Блоги» со всеми выбранными постами/блогами текущего пользователя
Настройки		
№	Проверка	Результат
37	Запуск страницы	Отображается меню из трёх пунктов: - изменение профиля - изменение пароля - удаление аккаунта
38	Запуск страницы изменения профиля	Во все поля для ввода подставляется корректная информация о пользователе и отображается фото профиля
39	Выбор новой фотографии и нажатие на кнопку сохранить	Фотография профиля изменяется и осуществляется переход на страницу профиля
40	Ввести корректные данные для изменения профиля	Информация о пользователе изменяется и осуществляется переход на страницу профиля
41	Не ввести данные в поле для изменения профиля	Под полями «Логин», «Email», «Имя» появляются соответствующие ошибки (остальные поля не обязательные для заполнения)
42	Ввести «Логин» или «Email», который уже используется другим пользователем	Появляется ошибка «Данное значение уже существует» под соответствующим полем
43	Ввести текст длиной 1 символ в поля для изменения профиля	Появляются ошибки под полями: - логин и имя: «Должно быть длиннее 4 и короче 32 символов» - страна и город: «Должно быть длиннее 2 и короче 64 символов»
44	Указать корректные данные при изменении пароля	Пароль успешно изменяется и осуществляется выход из приложения и переход на страницу авторизации
45	Ввести пароль короче 4 или длиннее 32 символов	Появляется ошибка «Должно быть длиннее 4 и короче 32 символов»
46	Значения в поле «Подтвердите пароль» и «Пароль» не совпадают	Появляется ошибка «Пароли не совпадают»
47	Ввести некорректный старый пароль	Появляется ошибка «Неверный пароль»
48	Нажать на пункт меню удаления аккаунта	Появляется вопрос «Вы уверены что хотите удалить аккаунт?» и возможность выбрать кнопки «Отмена» и «Удалить» При выборе «Удалить» - аккаунт удаляется и осуществляется переход на страницу авторизации

Изм.	Лист	№ докум.	Подпись	Дата

В таблице 2 приведён чек-лист для проведения тестирования клиента сотрудника/администратора.

Таблица 2 - Чек-лист «Тестирование клиента сотрудника/администратора»

Общие		
№	Проверка	Результат
1	Запуск программы	Для неавторизованного пользователя отображается окно авторизации, иначе страница с жалобами
2	Тестирование главного меню приложения	Для неавторизованного пользователя отображается только авторизация, иначе меню состоит из следующих элементов: жалобы, профиль, выход. Для администратора также есть: базовые жалобы, категории, сотрудники.  При нажатии на каждый элемент открывается соответствующая страницы, а при нажатии на «Выход» происходит соответственно выход из приложения и переход на страницу авторизации
Авторизация		
№	Проверка	Результат
3	Ввести данные в поля «Логин/Email» и «Пароль», которые существуют в БД, и нажать на кнопку «Войти»	Переход на страницу с жалобами
4	Ввод данных в поля «Логин/Email» и «Пароль», которые не существуют в БД	Появляется ошибка «Неправильный логин или пароль»
5	Не ввести данные в какое-либо из полей	Появляется ошибка «Поле обязательное для заполнения» под соответствующим полем
Профиль		
№	Проверка	Результат
6	Запуск страницы	Отображается полная информация о профиле пользователя и две кнопки: «Изменить» и «Изменить пароль». При нажатии на каждую из кнопок осуществляется переход на соответствующие страницы

Продолжение таблицы 2 – Чек-лист «Тестиирование клиента  
сотрудника/администратора»

№	Проверка	Результат
7	Запуск страницы изменения профиля	Во все поля для ввода подставляется корректная информация о пользователе. Поле «Роль» только для чтения (пользователь не может её изменить)
8	Ввести корректные данные для изменения профиля	Информация о пользователе изменяется и осуществляется переход на страницу профиля
9	Не ввести данные в какое-либо из полей для изменения профиля	Появляется ошибка «Поле обязательное для заполнения» под соответствующим полем
10	Ввести «Логин» или «Email», который уже используется другим пользователем	Появляется ошибка «Данное значение уже существует» под соответствующим полем
11	Ввести текст длиной 1 символ в поле для логина	Появляются ошибки «Должно быть длиннее 4 и короче 32 символов» под соответствующим полем
12	Указать корректные данные при изменении пароля	Пароль успешно изменяется и осуществляется выход из приложения и переход на страницу авторизации
13	Ввести пароль короче 4 или длиннее 32 символов	Появляется ошибка «Должно быть длиннее 4 и короче 32 символов»
14	Значения в поле «Подтвердите пароль» и «Пароль» не совпадают	Появляется ошибка «Пароли не совпадают»
15	Ввести некорректный старый пароль	Появляется ошибка «Неверный пароль»

**Жалобы**

№	Проверка	Результат
16	Запуск страницы	Появляется список фотографов с количеством жалоб, которые были поданы на них, с кнопкой перехода к подробной информации
17	Нажатие на кнопку «Подробнее» у одного из фотографов	Переход на страницу с таблицей постов/блогов фотографа с количеством жалоб, которые были поданы на них, и с кнопкой перехода к подробной информации  Также есть кнопка «Заблокировать пользователя»
18	Нажатие на кнопку «Заблокировать пользователя»	Переход на страницу списка фотографов с количеством жалоб, где больше не отображается заблокированный фотограф (так как все его жалобы автоматически обработаны)

Продолжение таблицы 2 – Чек-лист «Тестиирование клиента  
сотрудника/администратора»

№	Проверка	Результат
19	Нажатие на кнопку «Подробнее» у одного из постов/блогов фотографа	Переход на страницу с предпросмотром поста/блога и со списком всех жалоб на данный контент. Также есть кнопка «Отклонить» у каждой жалобы, кнопка «Отклонить все» и кнопка «Заблокировать контент»
20	Нажатие на кнопку «Отклонить»	Появляется надпись «Отклонена» вместо кнопки
21	Нажатие на кнопку «Отклонить все»	Переход на страницу с таблицей постов/блогов фотографа с количеством жалоб, где больше не отображается пост/блог, все жалобы которого были отклонены
22	Нажатие на кнопку «Заблокировать контент»	Переход на страницу с таблицей постов/блогов фотографа с количеством жалоб, где больше не отображается пост/блог, который был заблокирован
<b>Базовые жалобы</b>		
№	Проверка	Результат
23	Запуск страницы	Отображается список базовых жалоб с кнопками перехода для изменения и удаления. Также есть кнопка для добавления базовой жалобы
24	Нажатие на кнопку «Добавить»	Переход на страницу добавления базовой жалобы
25	Корректное заполнение полей для добавления базовой жалобы	Переход на страницу с подробной информацией о добавленной базовой жалобе
26	Нажатие на кнопку «Изменить» у базовой жалобы	Переход на страницу изменения базовой жалобы, где в поля для ввода подставляется корректная информация о базовой жалобе
27	Корректное заполнение полей для изменения базовой жалобы	Переход на страницу с подробной информацией об изменённой базовой жалобе
28	Не заполнить поле «Название» при добавлении/изменении базовой жалобы	Появляется ошибка «Поле обязательное для заполнения»
29	Ввести в поле «Название» при добавлении/изменении базовой жалобы название, которое уже используется в систему	Появляется ошибка «Данное значение уже существует»

Продолжение таблицы 2 – Чек-лист «Тестиирование клиента  
сотрудника/администратора»

№	Проверка	Результат
30	Ввести менее 4 или более 64 символов в поле «Название» при добавлении/изменении базовой жалобы	Появляется ошибка «Должно быть длиннее 4 и короче 64 символов»
31	Нажатие на кнопку «Удалить» у базовой жалобы	Переход на страницу с подтверждением удаления базовой жалобы  Если эта базовая жалоба закреплена за какой-либо жалобой, то появляется сообщение «Нельзя удалить базовую жалобу, так как за ней закреплены жалобы» и не показывается кнопка «Удалить»
32	Нажатие на кнопку «Удалить» при подтверждении	Переход на страницу со списком базовых жалоб, где нет удалённой базовой жалобы
Категории		
№	Проверка	Результат
33	Запуск страницы	Отображается список папок категорий и категорий в них с кнопками перехода для изменения и удаления. Также есть кнопка для добавления категории и добавления папки для категории
34	Нажатие на кнопку «Добавить»	Переход на страницу добавления категории, где в выпадающем списке находится список всех папок для категорий
35	Корректное заполнение полей для добавления категории	Переход на страницу с подробной информацией о добавленной категории
36	Нажатие на кнопку «Добавить папку»	Переход на страницу добавления папки
37	Корректное заполнение полей для добавления папки для категории	Переход на страницу с подробной информацией о добавленной папки для категории
38	Нажатие на кнопку «Изменить» у папки для категорий	Переход на страницу изменения папки, где в поля для ввода подставляется корректная информация о папки для категории
39	Корректное заполнение полей для изменения папки для категории	Переход на страницу с подробной информацией об изменённой папки для категории
40	Нажатие на кнопку «Изменить» у категории	Переход на страницу изменения папки, где в поля для ввода подставляется корректная информация о папки для категории

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 2 – Чек-лист «Тестиирование клиента  
сотрудника/администратора»

№	Проверка	Результат
41	Корректное заполнение полей для изменения категории	Переход на страницу с подробной информацией об изменённой категории
42	Не заполнить поле «Название» при добавлении/изменении категории или папки для категории	Появляется ошибка «Поле обязательное для заполнения»
43	Ввести в поле «Название» при добавлении/изменении категории или папки для категории название, которое уже используется в систему	Появляется ошибка «Данное значение уже существует»
44	Ввести менее 4 или более 64 символов в поле «Название» при добавлении/изменении категории или папки для категории	Появляется ошибка «Должно быть длиннее 4 и короче 64 символов»
45	Нажатие на кнопку «Удалить» у папки для категории	Переход на страницу с подтверждением удаления папки для категорий  Если в этой папки есть категории, то появляется сообщение «Нельзя удалить папку, так как за ней закреплены категории» и не показывается кнопка «Удалить»
46	Нажатие на кнопку «Удалить» у категории	Переход на страницу с подтверждением удаления категории  Если эта категория закреплена за каким-либо контентом, то появляется сообщение «Нельзя удалить категорию, так как за ней закреплён контент» и не показывается кнопка «Удалить»
47	Нажатие на кнопку «Удалить» при подтверждении	Переход на страницу со списком категорий, где нет удалённой категории/папки для категорий
<b>Сотрудники</b>		
№	Проверка	Результат
48	Запуск страницы	Отображается список сотрудников с кнопками перехода для изменения и удаления. Также есть кнопка для добавления сотрудника
49	Нажатие на кнопку «Добавить»	Переход на страницу добавления сотрудника, где в выпадающем списке находится список всех ролей пользователя

Изм.	Лист	№ докум.	Подпись	Дата

Продолжение таблицы 2 – Чек-лист «Тестирование клиента  
сотрудника/администратора»

№	Проверка	Результат
50	Корректное заполнение полей для добавления сотрудника	Переход на страницу с подробной информацией о добавленном сотруднике
51	Нажатие на кнопку «Изменить» у сотрудника	Переход на страницу изменения сотрудника, где в поля для ввода подставляется корректная информация о выбранном сотруднике
52	Корректное заполнение полей для изменения сотрудника	Переход на страницу с подробной информацией об изменённом сотруднике
53	Не ввести данные в какое-либо из полей для добавления/изменения сотрудника	Появляется ошибка «Поле обязательное для заполнения» под соответствующим полем
54	Ввести менее 4 или более 32 символов в поле «Логин» при добавлении/изменении сотрудника	Появляется ошибка «Должно быть длиннее 4 и короче 32 символов»
55	Ввести пароль короче 4 или длиннее 32 символов при добавлении сотрудника	Появляется ошибка «Должно быть длиннее 4 и короче 32 символов»
56	Значения в поле «Подтвердите пароль» и «Пароль» не совпадают при добавлении сотрудника	Появляется ошибка «Пароли не совпадают»
57	Нажатие на кнопку «Удалить» у сотрудника	Переход на страницу с подтверждением удаления сотрудника.  При нажатии на кнопку «Удалить» происходит переход на страницу со списком сотрудников, где нет удалённого сотрудника

Система, на которой проводилось тестирование:

- ОС: Windows 10 Корпоративная 64-разрядная (10.0, сборка 19045);
- процессор: Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz, 2300 МГц, ядер: 2;
- язык системы: русский;
- оперативная память (RAM): 8,0 ГБ;
- в качестве web браузера использовался Google Chrome - версия 114.0.5735.110.

Изм.	Лист	№ докум.	Подпись	Дата

## Заключение

В данной бакалаврской работе в соответствии с заданием был разработан web-сайт «Социальная сеть для фотографов».

В ходе работы были выполнены следующие задачи:

- проанализирована предметная область и выявлены требования к программе;
- спроектирована система;
- создана база данных;
- разработано серверное приложение;
- разработаны два типа клиентских приложений: для пользователя и для сотрудника/администратора;
- осуществлено его тестирование.

Реализованное приложение соответствует всем заявленным требованиям, которые были сформулированы в анализе технического задания, при этом расширяя их и добавляя дополнительные возможности.

На основе проведенного тестирования было подтверждено, что весь функционал системы работает корректно и соответствует требованиям.

Таким образом, можно сделать вывод о том, что разработанное приложение «Социальная сеть для фотографов» является эффективным и удобным инструментом для организации работы фотографов, позволяющим им делиться своими работами и следить за другими фотографами.

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

## Список используемых источников

1. Аналог 500px: [Электронный ресурс] // URL: <https://500px.com/> (Дата обращения – 20.03.2023)
2. Аналог Flickr: [Электронный ресурс] // URL: <https://www.flickr.com/> (Дата обращения – 20.03.2023)
3. Аналог Habr: [Электронный ресурс] // URL: <https://habr.com/ru/all/> (Дата обращения – 20.03.2023)
4. Библиотека аутентификации Microsoft Identity Web: [Электронный ресурс] // URL: <https://habr.com/ru/all/> (Дата обращения – 17.04.2023)
5. Обзор Entity Framework: [Электронный ресурс] // URL: <https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/ef/overview> (Дата обращения – 25.03.2023)
6. Руководство по ASP.NET Core: [Электронный ресурс] // URL: <https://metanit.com/sharp/aspnet6/> (Дата обращения – 10.04.2023)
7. Руководство по Axios API: [Электронный ресурс] // URL: [https://axios-http.com/ru/docs/api\\_intro](https://axios-http.com/ru/docs/api_intro) (Дата обращения – 1.05.2023)
8. Руководство по Bootstrap 5: [Электронный ресурс] // URL: <https://getbootstrap.com/docs/5.0/getting-started/introduction/> (Дата обращения – 25.04.2023)
9. Руководство по Flexbox CSS: [Электронный ресурс] // URL: <https://tpverstak.ru/flex-cheatsheet/> (Дата обращения – 28.04.2023)
10. Руководство по WEB API ASP.NET Core: [Электронный ресурс] // URL: <https://metanit.com/sharp/aspnet5/23.1.php> (Дата обращения – 09.04.2023)

Изм.	Лист	№ докум.	Подпись	Дата

МИВУ 09.03.04-8.000 ВКР

Лист

77

## Приложение А. Снимки окон программы

The left screenshot shows the 'Registration' page. It features a sidebar with icons for user profile, home, search, and settings. The main area has a title 'Регистрация' and fields for 'Логин' (user8), 'Email' (user8@mail.ru), 'Пароль' (\*\*\*\*\*), and 'Подтвердите пароль' (\*\*\*\*\*). A 'Зарегистрироваться' button is at the bottom.

The right screenshot shows the 'Authorization' page. It has a similar sidebar. The main area has a title 'Авторизация' and fields for 'Логин / Email' (user8) and 'Пароль' (\*\*\*\*\*). A 'Войти' button is at the bottom.

Рисунок 41 – страница регистрации и авторизации

The left screenshot shows the user profile page for 'user8'. It displays a profile picture of a woman, the username 'user8', and the name 'Ирина Калмыкова'. Below this are tabs for 'Подробне' (selected), 'Посты', and 'Блоги'. To the left is a sidebar with icons for user profile, home, search, and settings. Below the sidebar are sections for 'Подписчики' (3) and 'Подписки' (4), each with a 'View' link. The 'Описание' section contains a bio about being a famous children's photographer. At the bottom are 'Страна: Россия' and 'Город: Москва'.

The right screenshot shows the same profile page, but the 'Посты' tab is selected. It displays a post by 'user8' from '15.06.2023 10:52:26' showing a landscape photo of a field with purple flowers and a city in the background, with the caption 'Природа'.

Рисунок 42 – профиль текущего пользователя

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

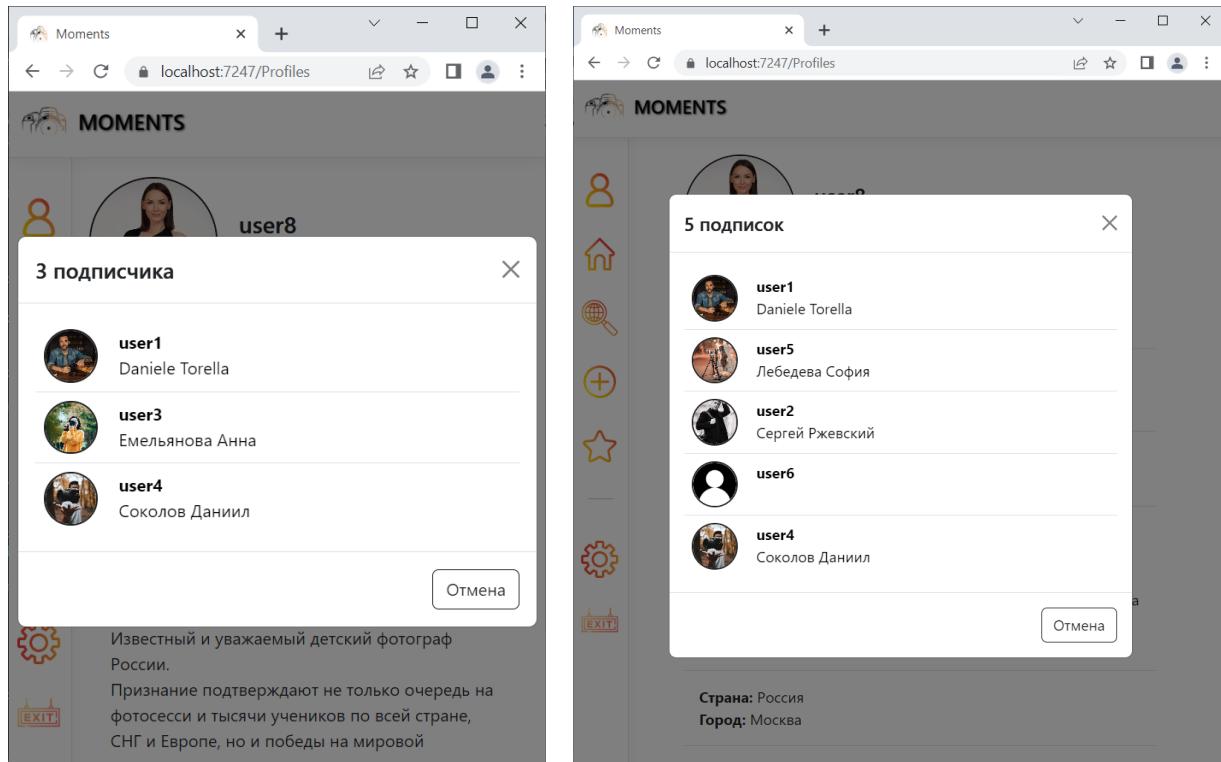


Рисунок 43 – список подписчиков и подписок

Рисунок 44 – профиль другого пользователя

Изм.	Лист	№ докум.	Подпись	Дата

МИВУ 09.03.04-8.000 ВКР

Лист

79

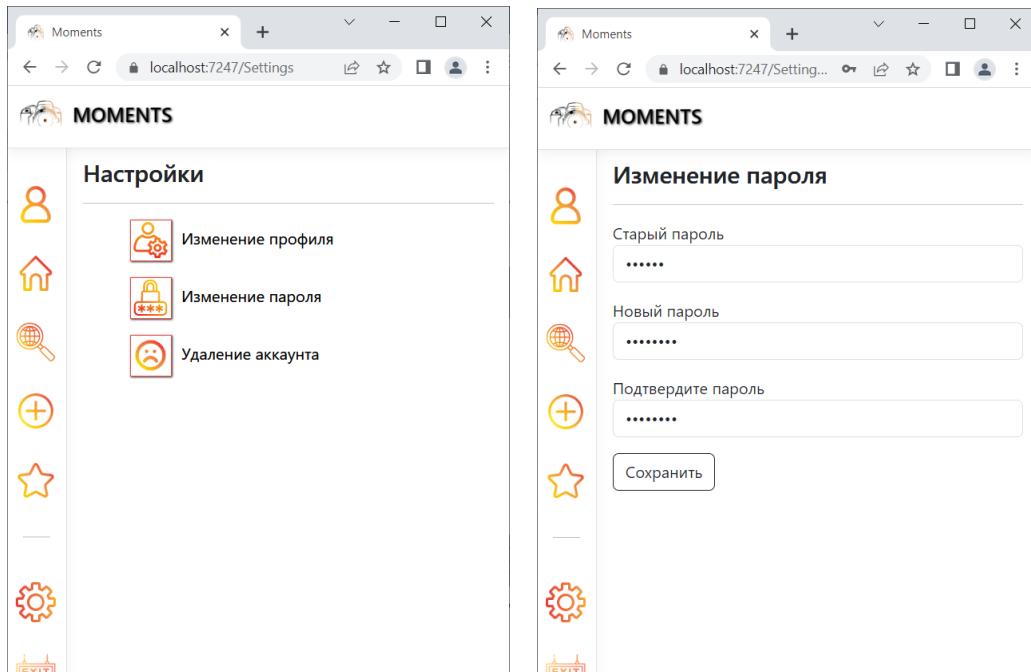


Рисунок 45 – страница настроек и страница изменения пароля

Рисунок 46 – изменение профиля пользователя

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

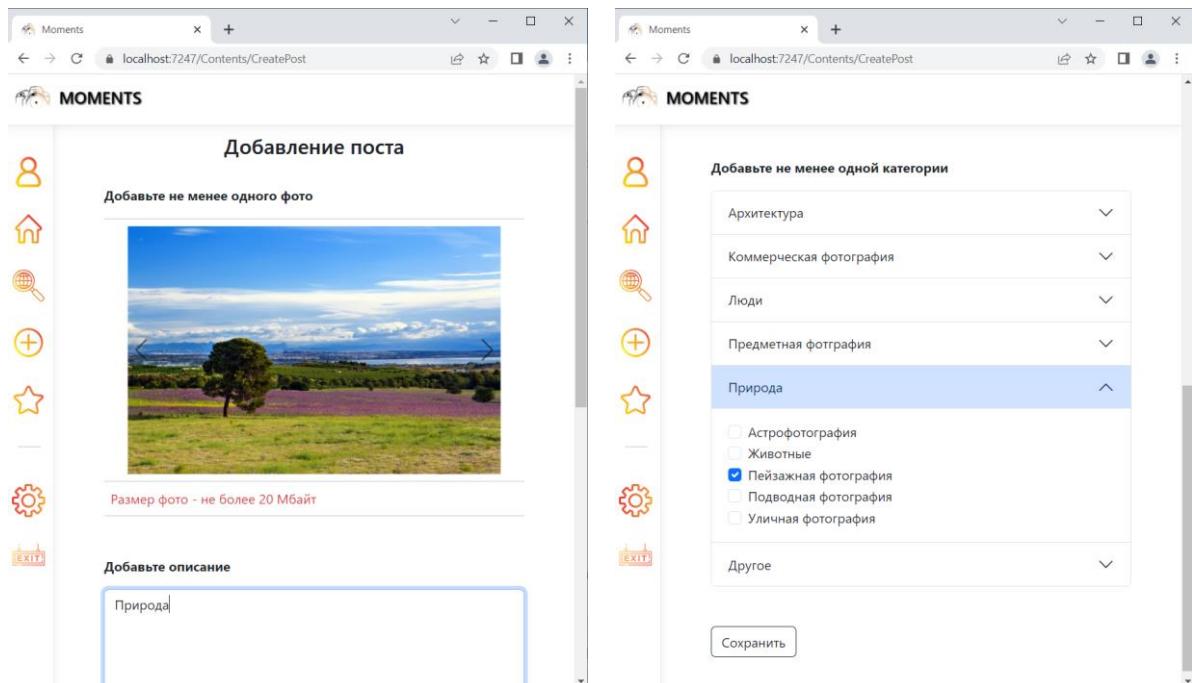


Рисунок 47 – добавления поста

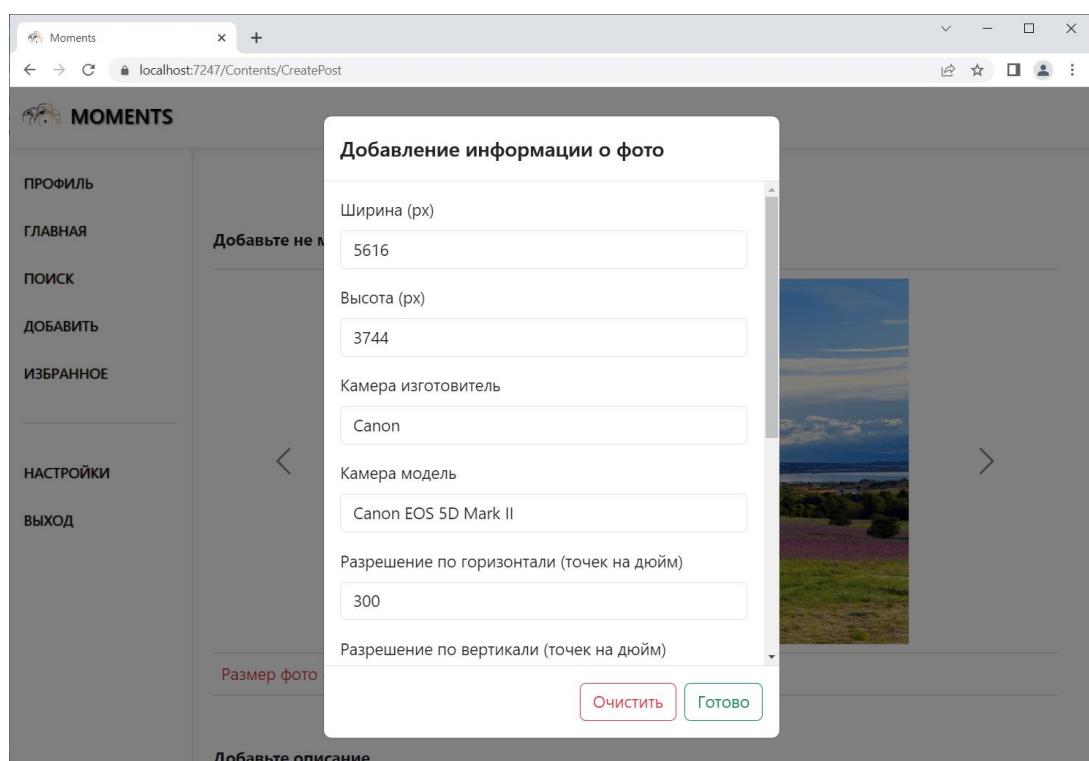


Рисунок 48 – добавление информации о фотографии

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

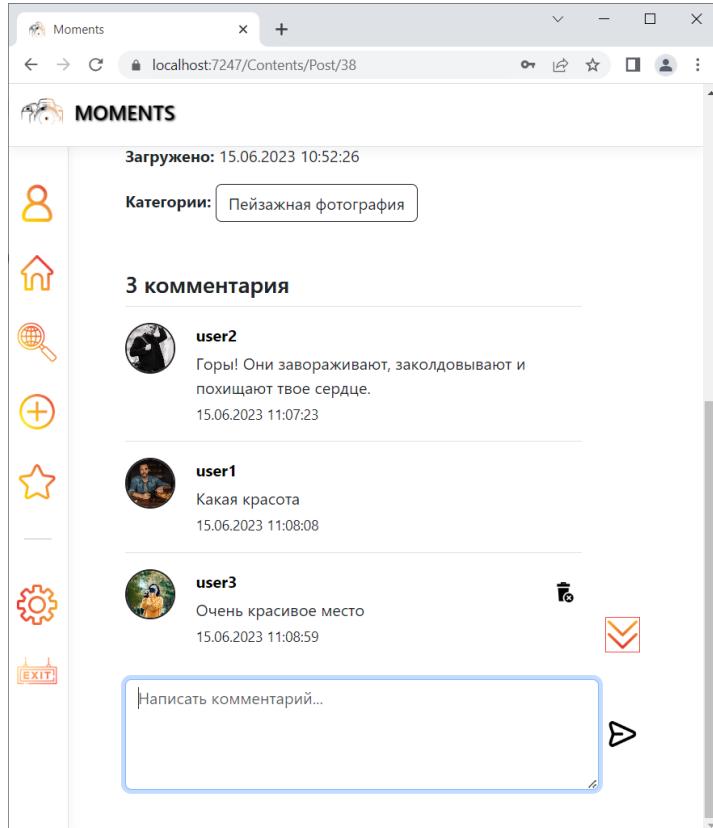


Рисунок 49 – комментарии к посту/блогу

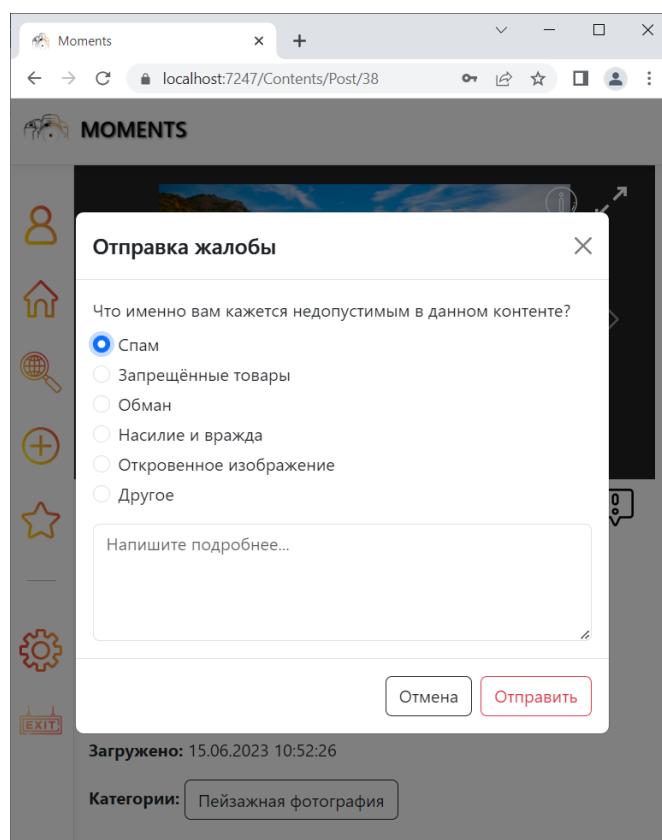


Рисунок 50 – окно для отправки жалобы

Изм.	Лист	№ докум.	Подпись	Дата

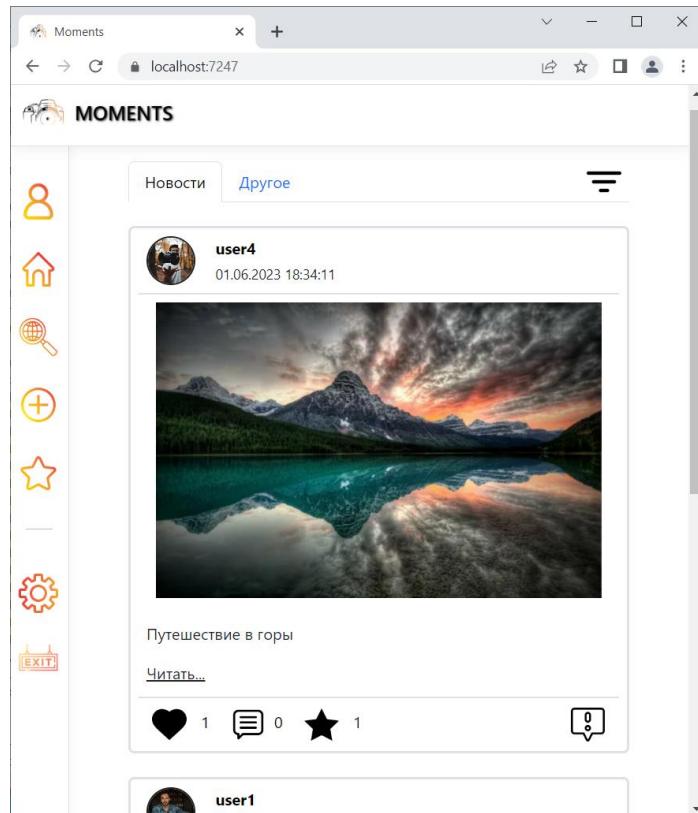


Рисунок 51 – главная страница

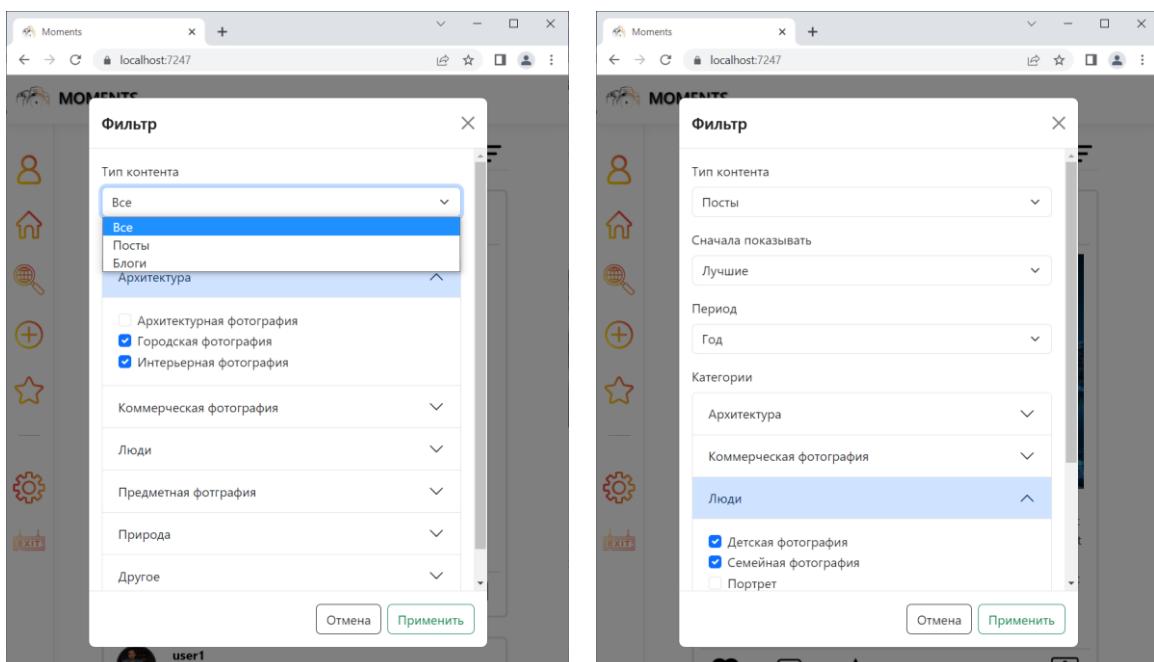


Рисунок 52 – фильтры на странице новостей (для вкладки «Новости») и для вкладки «Другое»)

Изм.	Лист	№ докум.	Подпись	Дата

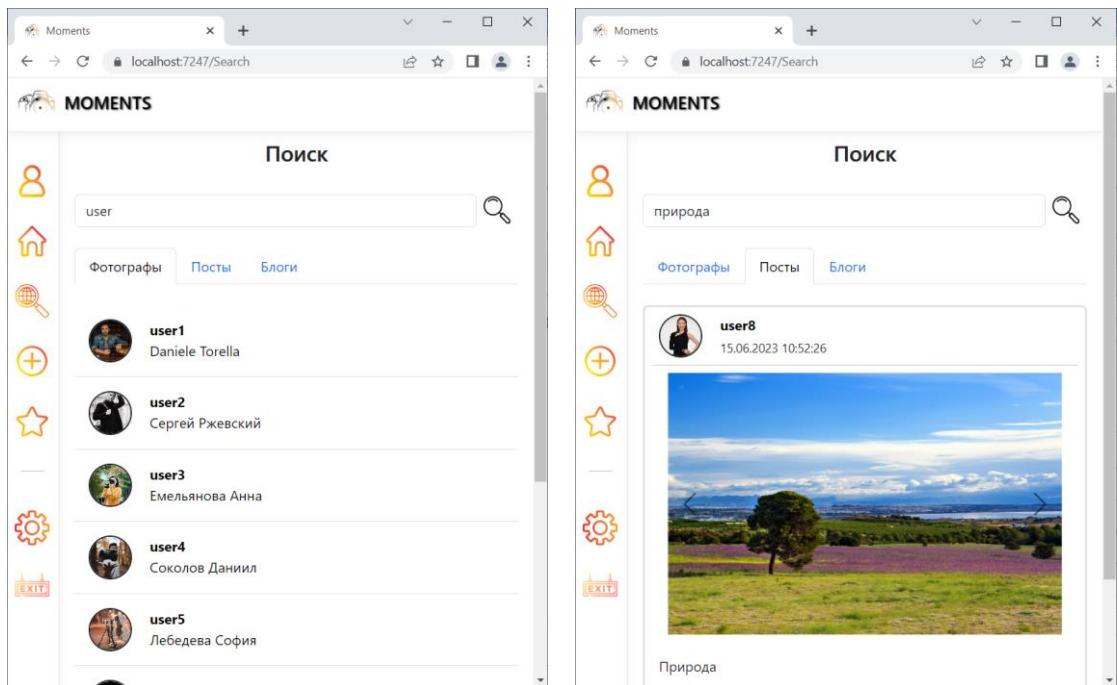


Рисунок 53 – страница поиска

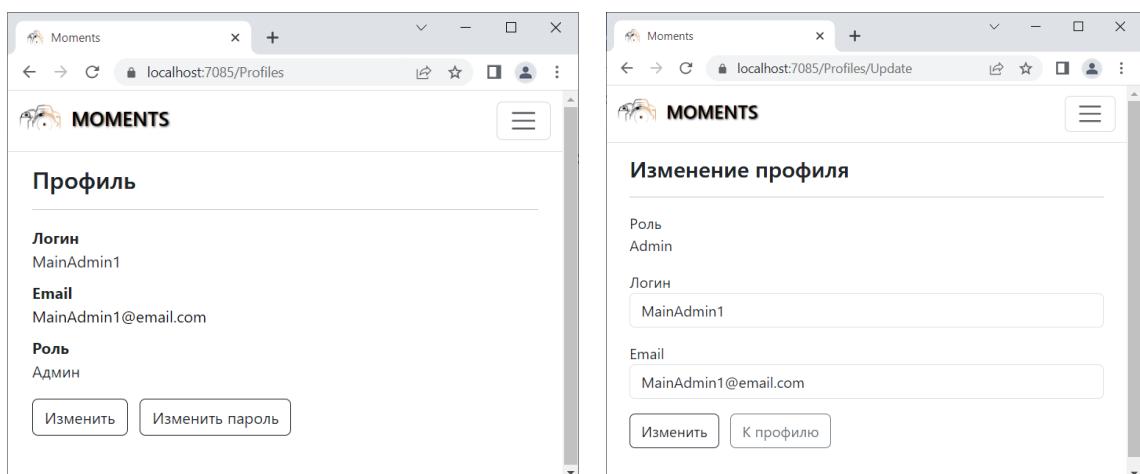


Рисунок 54 – профиль сотрудника/администратора и его изменение

#	Логин пользователя	Количество жалоб	
1	user1	9	Подробнее
2	user4	3	Подробнее
3	user8	3	Подробнее

Рисунок 55 – список количества жалоб на фотографов

Изм.	Лист	№ докум.	Подпись	Дата

The screenshot shows a web browser window titled 'Moments' at the URL 'localhost:7085/Complaints/Photographer/1'. The page header includes the 'MOMENTS' logo, navigation links for 'Жалобы' (Complaints) and 'Компоненты' (Components), and user account links for 'Профиль' (Profile) and 'Выйти' (Logout). The main content area is titled 'Жалобы на пользователя' (Complaints about the user). It displays a circular profile picture of a man with a beard, labeled 'user1' and 'Daniele Torella'. Below the profile are two buttons: 'Заблокировать пользователя' (Block user) in red and 'К списку' (To list) in grey. A table lists three posts with their complaint counts and 'Подробнее' (More details) buttons:

#	Контент	Количество жалоб	
1	Пост №3	3	<a href="#">Подробнее</a>
2	Пост №9	2	<a href="#">Подробнее</a>
3	Блог №10	4	<a href="#">Подробнее</a>

Рисунок 56 – список количества жалоб на посты/блоги фотографа

The screenshot shows a web browser window titled 'Moments' at the URL 'localhost:7085/Complaints/Content/20'. The page header is identical to the previous one. The main content area shows a large image of a landscape with mountains and clouds. Below the image is the text 'Путешествие в горы' and a link 'Читать...'. At the bottom of the page are four buttons: 'Отклонить все' (Reject all) in green, 'Заблокировать контент' (Block content) in red, 'К списку' (To list) in grey, and 'К пользователю' (To user) in grey. A table lists three complaints with 'Отклонить' (Reject) buttons:

#	Базовая жалоба	Доп. описание	
1	Запрещённые товары		<a href="#">Отклонить</a>
2	Запрещённые товары	Срочно заблокируйте пользователя	<a href="#">Отклонить</a>
3	Обман		<a href="#">Отклонить</a>

Рисунок 57 – список жалоб на пост/блог

Изм.	Лист	№ докум.	Подпись	Дата
------	------	----------	---------	------

МИВУ 09.03.04-8.000 ВКР

The screenshot shows two browser windows side-by-side. The left window displays a list titled 'Базовые жалобы' (Basic Complaints) with the following items:

#	Название	Изменить	Удалить
1	Спам	Изменить	Удалить
2	Запрещённые товары	Изменить	Удалить
3	Обман	Изменить	Удалить
4	Насилие и вражда	Изменить	Удалить
5	Откровенное изображение	Изменить	Удалить
6	Другое	Изменить	Удалить

A green 'Добавить' (Add) button is located at the top left of the list. The right window shows a form titled 'Добавление базовой жалобы' (Adding a basic complaint) with a single input field labeled 'Название' containing the value 'Спам'. It also features a green 'Добавить' button and a blue 'К списку' (To list) button.

Рисунок 58 – список базовых жалоб и её добавление

The screenshot shows two browser windows side-by-side. The left window displays a form titled 'Изменение базовой жалобы' (Editing a basic complaint) with a single input field labeled 'Название' containing the value 'Спам'. It includes a green 'Изменить' (Edit) button and a blue 'К списку' (To list) button. The right window shows a form titled 'Удаление базовой жалобы' (Deleting a basic complaint) with a single input field labeled 'Название' containing the value 'Спам'. It features a red 'Удалить' (Delete) button and a blue 'К списку' (To list) button.

Рисунок 59 – изменение и удаление базовой жалобы

Изм.	Лист	№ докум.	Подпись	Дата

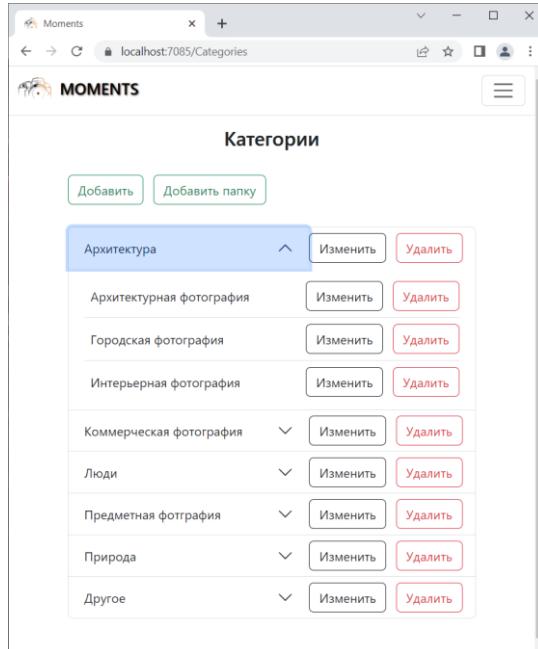


Рисунок 60 – список папок с категориями

Рисунок 61 – добавление папки для категории и категории

Рисунок 62 – подробная информация папки для категории и категории

Изм.	Лист	№ докум.	Подпись	Дата

The left screenshot shows the 'Изменение папки для категории' (Change folder for category) form. It has a dropdown for 'CategoryDirId' (set to 'Люди') and a text input for 'Название' (set to 'Астрофотография'). Buttons include 'Изменить' (Change) and 'К списку' (To list).

The right screenshot shows the 'Изменение категории' (Change category) form. It has a dropdown for 'CategoryDirId' (set to 'Природа') and a text input for 'Название' (set to 'Астрофотография'). Buttons include 'Изменить' (Change) and 'К списку' (To list).

Рисунок 63 – изменение информации о папке для категории и категории

The left screenshot shows the 'Удаление папки для категории' (Delete folder for category) form. It displays a warning message: 'Нельзя удалить папку, так как за ней закреплены категории' (Cannot delete the folder, as it is attached to categories). It includes a dropdown for 'Название папки' (set to 'Люди') and a 'К списку' (To list) button.

The right screenshot shows the 'Удаление категории' (Delete category) form. It lists 'Папка' (Folder) 'Природа' and 'Название' (Name) 'Астрофотография'. It includes a 'Удалить' (Delete) button and a 'К списку' (To list) button.

Рисунок 64 – удаление папки для категории и категории

The screenshot shows a table titled 'Сотрудники' (Employees) with columns: '#', 'Логин' (Login), 'Email', and 'Роль' (Role). The data is as follows:

#	Логин	Email	Роль		
1	MainAdmin2	MainAdmin2@email.com	Админ	<a href="#">Изменить</a>	<a href="#">Удалить</a>
2	MainAdmin1	MainAdmin1@email.com	Админ	<a href="#">Изменить</a>	<a href="#">Удалить</a>
3	MainAdmin3	MainAdmin3@mail.com	Админ	<a href="#">Изменить</a>	<a href="#">Удалить</a>
4	Employee4	Employee4@mail.com	Сотрудник	<a href="#">Изменить</a>	<a href="#">Удалить</a>
5	Employee1	employee1@example.com	Сотрудник	<a href="#">Изменить</a>	<a href="#">Удалить</a>
6	Employee2	employee2@example.com	Сотрудник	<a href="#">Изменить</a>	<a href="#">Удалить</a>
7	Employee3	Employee3@email.com	Сотрудник	<a href="#">Изменить</a>	<a href="#">Удалить</a>

Рисунок 65 – список сотрудников

Изм.	Лист	№ докум.	Подпись	Дата

МИВУ 09.03.04-8.000 ВКР

Моменты

Жалобы Компоненты

Профиль Выйти

### Добавление сотрудника

Роль  
Админ

Логин  
Employee5

Email  
Employee5@email.com

Пароль  
.....

Подтвердите пароль  
.....

**Добавить** **К списку**

Рисунок 66 – добавление сотрудника

Моменты

Жалобы Компоненты

### Изменение сотрудника

Роль  
Сотрудник

Логин  
Employee5

Email  
Employee5@email.com

**Изменить** **К списку**

### Удаление сотрудника

Логин  
Employee5

Email  
Employee5@email.com

Роль  
Админ

**Удалить** **К списку**

Рисунок 67 – изменение и удаление сотрудника

Изм.	Лист	№ докум.	Подпись	Дата

## Приложение Б. Исходный код ПО

Полной код разработанного приложения «Социальная сеть фотографов»  
представлен по ссылке: <https://github.com/vivir-paravolar/WebAppNetworkForPhotographers>.

Также можно воспользоваться QR кодом:



Изм.	Лист	№ докум.	Подпись	Дата

МИВУ 09.03.04-8.000 ВКР

Лист  
90