МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационные системы и технологии

Специальность 1-98 01 03 “Программное обеспечение информационной

безопасности мобильных систем”

Специализация Инженер-программист

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«Социальная сеть для обмена фото “Instaclone”»

Выполнил студент Першай Ян Борисович

(Ф.И.О.)

Руководитель проекта ст. преподаватель Блинова Е.А. (учен. степень, звание, должность, Ф.И.О., подпись)

Заведующий кафедрой к.т.н., доц. Смелов В.В .

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

Консультант: ст. преподаватель Блинова Е.А. (учен. степень, звание, должность, Ф.И.О., подпись)

Нормоконтролер: ст. преподаватель Блинова Е.А (учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

Минск 2020

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc41229094)

[1 Обзор аналогов приложений «Instaclone» 4](#_Toc41229095)

[1.1 Приложение «Instagram» 4](#_Toc41229096)

[1.2 Приложение «VSCO» 5](#_Toc41229097)

[1.3 Выводы по главе «Обзор аналогов приложений «Instaclone»» 6](#_Toc41229099)

[2 Проектирование программного продукта 7](#_Toc41229100)

[2.1 Проектирование мобильного приложения 7](#_Toc41229101)

[2.2 Проектирование базы данных 9](#_Toc41229102)

[2.3 Проектирование сервера 1](#_Toc41229103)1

[2.4 Выводы по главе «Проектирование программного продукта» 1](#_Toc41229104)1

[3 Программная реализация приложений 1](#_Toc41229105)2

[3.1 Технические средства для разработки 1](#_Toc41229106)2

[3.2 Разработка баз данных 1](#_Toc41229107)2

[3.3 Разработка мобильного приложения 13](#_Toc41229108)

[3.4 Разработка сервера 17](#_Toc41229109)

[3.5 Вывод по главе «Программная реализация приложений» 1](#_Toc41229110)9

[4 Руководство пользователя 20](#_Toc41229114)

[5.1 Выводы по главе «Руководство пользователя» 3](#_Toc41229115)0

[ЗАКЛЮЧЕНИЕ 3](#_Toc41229116)1

[Список использованных источников 3](#_Toc41229117)3

[ПРИЛОЖЕНИЕ А 34](#_Toc41229118)

[ПРИЛОЖЕНИЕ Б 35](#_Toc41229119)

[ПРИЛОЖЕНИЕ В 36](#_Toc41229120)

# **Введение**

Цель курсового проекта: предоставление пользователям возможности обмена фотографиями.

Задачи курсового проекта:

1. Изучение программных средств, аналогичных теме курсового проекта

Данная задача позволит изучить аналоги данному программному средству, разобраться в их минусах, плюсах. Найти отличия между аналогами и данным программным средством

1. Проектирование приложения

Проектирование приложения даёт возможность увидеть предварительную концепцию приложения, позволяет спланировать разработку программного средства и соответствующе распределить ресурсы и время, которые будут потрачены на разработку приложения.

1. Реализация программного средства

В курсовом проекте будет реализовано приложение для обмена фотографиями между пользователями. Данная концепция актуальна в век социальных сетей, где люди часто делятся кадрами из своей жизни с окружающими.

1. Тестирование разработанного продукта

Важный этап разработки, позволяющий проверить функциональность приложения и его реакцию на исключительные ситуации, например ввод данных неподходящего формата, неверные данные авторизации и др.

1. Разработка руководства пользователя

Данная задача подразумевает разработку руководства пользователя, призванное помочь пользователю разобраться с работой приложения и его основным функционалом. Это является важной частью разработки, поскольку определённые пользователи заинтересованы перед использованием ознакомиться с руководством пользователя, чтобы изучить функционал приложения.

# **1 Обзор аналогов приложений «Instaclone»**

# **1.1 Приложение «Instagram»**

Данное приложение на сегодняшний день является самым популярным программным средством для обмена фотографиями. Доступ к функционалу осуществляется только с доступом к интернету. Основная функция－ предоставление возможности пользователям обмениваться фото и оценивать их. Есть возможность публиковать как временные(на 24 часа) фото, так и постоянные.

Приложение предоставлено на скриншоте ниже (рис 1.1.1).

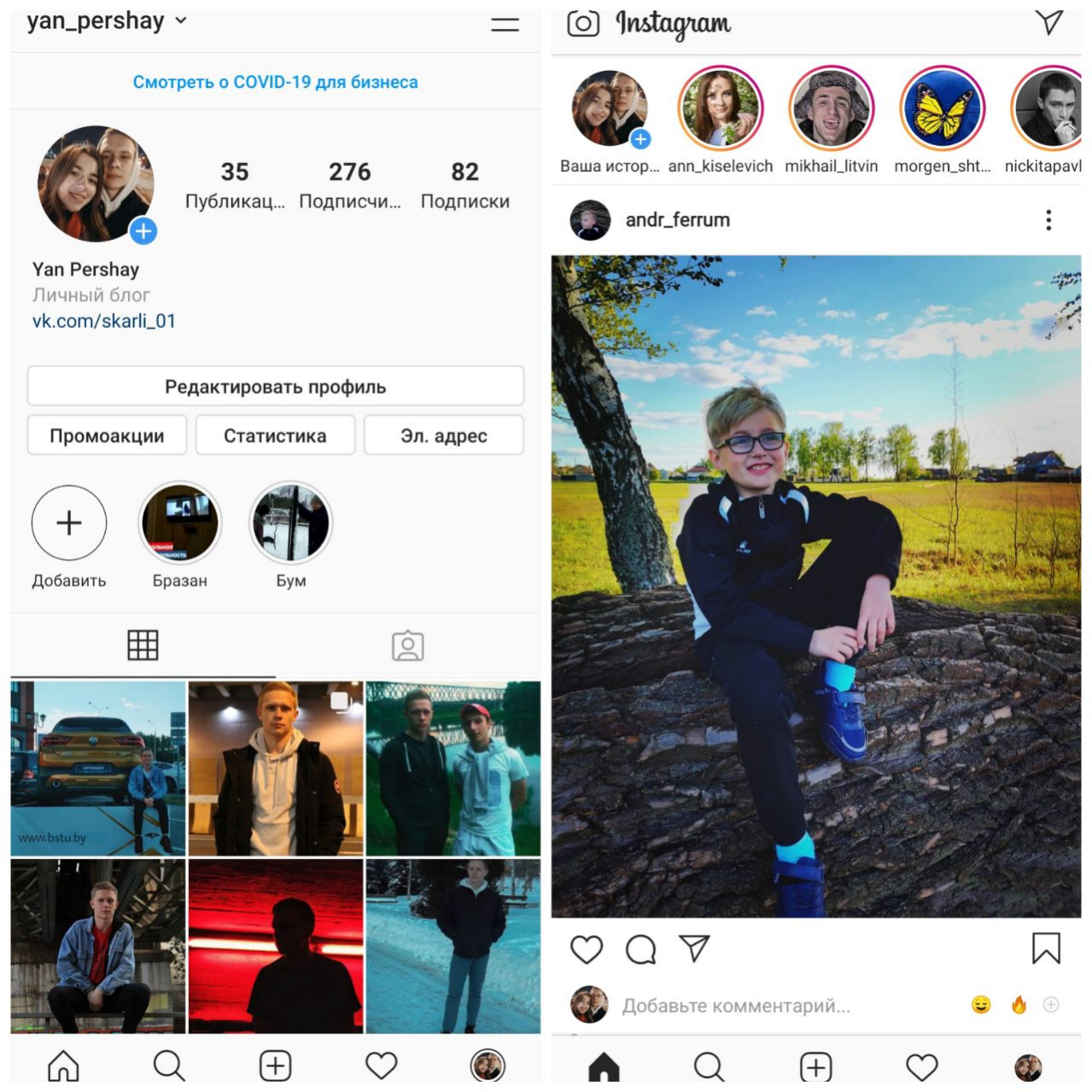


Рисунок 1.1.1 –Приложение «Instagram»

Также есть возможность оставлять комментарии, делиться видео, вести диалоги. Накладывать маски на фото.

# **1.2 Приложение «VSCO»**

В данном приложении можно не только делиться фото, но оно также имеет множество встроенных функций для редактирования фотографий, тонкой настройки снимка, возможность просмотра подборок ТОП-лучших фото за неделю.

Интерфейс приложения представлен на скриншоте ниже (рис 1.2.1).

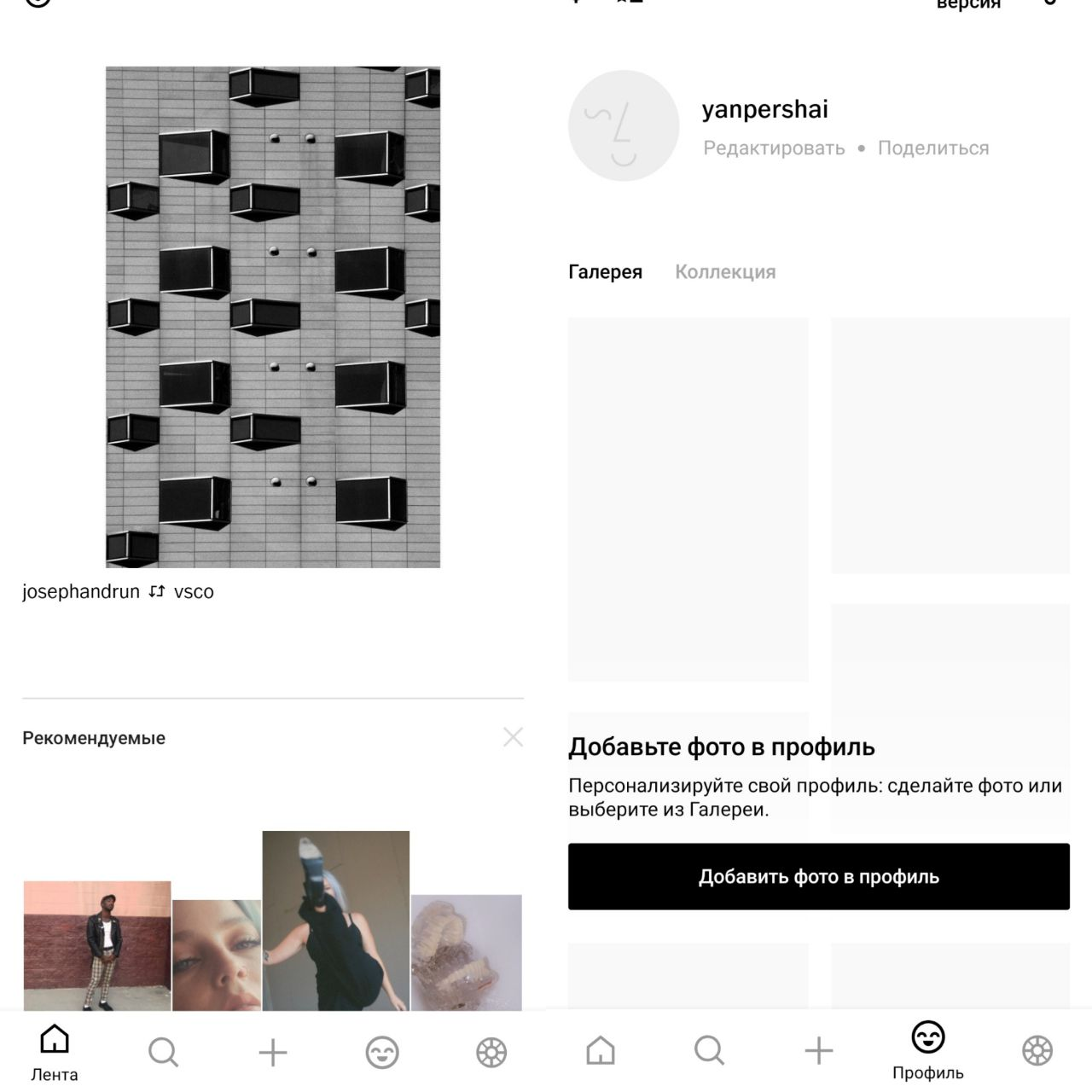


Рисунок 1.2.1 – Приложение «VSCO»

Лента имеет достаточно примитивный интерфейс т.к. основной задачей данного приложения является именно редактирование фото, наложение фильтров. Тем не менее, ввиду наличия возможности обмена фотографиями, VSCO является аналогом моего программного продукта.

# **1.3 Выводы по главе «Обзор аналогов приложений ”Instaclone”»**

В данной главе были показаны два наиболее популярных приложения, которые являются аналогами приложения «Instaclone»: «Instagram» и «VSCO». В результате подытожим сравнения.

“Instagram” является прототипом моего программного средства “Instaclone”. У него достаточно удобный и интуитивно понятный дизайн, хорошо оптимизированная работа. В отличие от прототипа, в моем программном средстве отсутствует система оценивания фото. Это можно обусловить тем, что в погоне за «лайками и подписками» люди фокусируются именно на статистике и общественном одобрении, а не на желании поделиться красивой фотографией, посмотреть на события жизни других пользователей.

К плюсам приложения “Instagram” относится наличие системы “Stories”, когда можно публиковать «истории» -- фото, исчезающие через 24 часа, что позволит гораздо чаще делиться своими событиями и не загромождать новостную ленту постами.

Также, большим плюсом является наличие системы переписок “Direct”, что мотивирует к использованию данного приложение в большем количестве случаев.

К плюсам приложения “VSCO” относится наличие возможности наложения фильтров на фото, тонкой настройки изображения, большой функционал редактирования фото.

К минусам относится примитивный и недостаточно интуитивно-понятный дизайн ленты и профиля, а также ограничение множества функционала платной подпиской.

# **2 Проектирование программного продукта**

# **2.1 Проектирование мобильного приложения**

В данной курсовой работе необходимо было разработать приложение, позволяющее пользователям осуществлять обмен фотографиями.

Если пользователь первый раз запускает приложение на устройстве или не авторизован, то открывается экран входа. У пользователя есть возможность ввести имя и пароль, если он зарегистрирован в приложении, либо зарегистрироваться.

При входе на страницу регистрации пользователь вводит свой email, придумывает имя пользователя, которым будет идентифицироваться в приложении, и придумывает пароль. Он регистрируется и ему предлагается ввести свои данные (имя, фамилия, возраст, город, информацию о себе). После сохранения данных, пользователь переходит на страницу авторизации.

В случае ввода верных логина и пароля пользователю открывается домашний экран, где отображены все публикации пользователей в новостной ленте.

На каждом экране снизу предоставляется нижнее меню, позволяющее перейти на главный экран, на экран добавления фото и на экран профиля.

Когда пользователь хочет добавить фото, сначала необходимо предоставить разрешения для доступа к внешней памяти, далее у него отображаются папки с картинками из внешней памяти. После выбора фото открывается экран, где пользователь придумывает подпись и публикует фотографию.

При выборе просмотра профиля, открывается экран, где отображается профиль пользователя и его фото. При нажатии на фото открывается публикация с этой фотографией и есть возможность удалить данную публикацию.

При выходе из аккаунта открывается экран входа.

При нажатии на фото профиля на публикации пользователя должен открываться экран, содержащий информацию о данном пользователе и отображающий все фото, опубликованные этим пользователем.

В случае выбора пользователем редактирования профиля, открывается экран, содержащий поля ввода с данными пользователя, которые можно изменить и сохранить изменения.

При нажатии на фотографию в своем профиле, она открывается в виде публикации, содержащей ник пользователя, описание к фото. На публикации есть иконка удаления, при нажатии на которую данная публикация удаляется.

Можно ознакомиться с UML-диаграммой, а именно диаграммой вариантов использования. Она отображает функциональность программного продукта с точки зрения получения значимого результата для пользователя. (рис. 2.1.1)

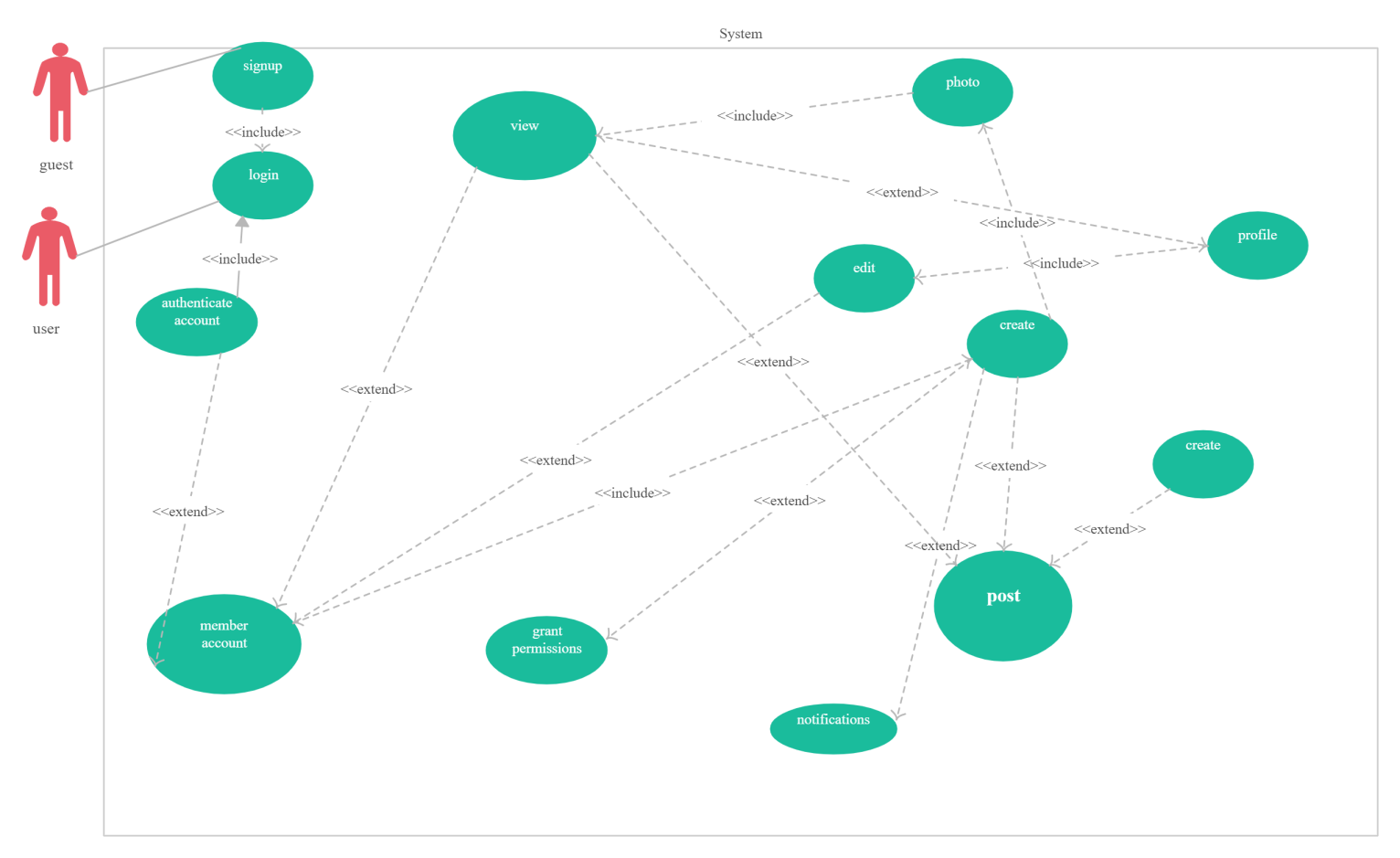


Рисунок 2.1.1 – Диаграмма вариантов использования

Таким образом приложение должно выполнять функции:

* Регистрация пользователя;
* Ввод данных пользователя;
* Авторизация пользователя;
* Просмотр публикаций других пользователей;
* Просмотр профилей других пользователей;
* Осуществлять доступ к директориям с фотографиями на устройстве;
* Добавление фото с описанием на сервер;
* Просмотр своего профиля;
* Редактирование своего профиля;
* Просмотр своей публикации;
* Удаление своей публикации.

# **2.2 Проектирование базы данных**

В базе данных данного приложения созданы для использования 3 таблицы. Скрипт создания БД и таблиц находится в Приложении Б. Ниже представлена их характеристика.

В таблице «Users» содержится информация для авторизации пользователей. Столбцы данной таблицы:

* UserId – уникальный идентификатор (Guid, создаётся на сервере), является первичным ключом;
* Username – строковый тип данных, максимальная длина 20, уникальное значение;
* Email – строковый тип данных, максимальная длина 20;
* Password – строковый тип данных, максимальной длины, хешируется на сервере(Приложение В);

В таблице «UserData» содержится информация о пользователях. Столбцы данной таблицы:

* IdData – целочисленный тип данных, первичный ключ;
* UserId – уникальный идентификатор, вторичный ключ, ссылается на первичный ключ таблицы Users, служит для связи этих таблиц;
* Firstname – содержит имя пользователя, строковый тип данных, максимальная длина 20, неуникальное значение, допускается null;
* Lastname – содержит фамилию пользователя, строковый тип данных, максимальная длина 20, неуникальное значение, допускается null;
* Age – содержит возраст пользователя, целочисленный тип данных, неуникальное значение, допускается null;
* Description – содержит дополнительную информацию о пользователе, строковый тип данных, максимальная длина 30, неуникальное значение, допускается null;
* City – содержит город пользователя, строковый тип данных, максимальная длина 20, неуникальное значение, допускается null;
* Followers – содержит количество подписчиков пользователя, целочисленный тип данных, неуникальное значение, не допускается null;
* Follows – содержит количество подписок пользователя, целочисленный тип данных, неуникальное значение, не допускается null;

В таблице «Posts» содержатся посты пользователей. Столбцы данной таблицы:

* PostId – целочисленный тип данных, первичный ключ;
* UserId – уникальный идентификатор, вторичный ключ, ссылается на первичный ключ таблицы Users, служит для связи этих таблиц;
* Caption – содержит описание к фото, строковый тип данных, максимальная длина 40, неуникальное значение, допускается null;
* Image – содержит в себе фото в строковом формате, имеет максимальную длину, не допускается null;
* DateCreated – содержит в себе дату создания публикации, тип DateTime, не допускается null;

Схема базы данных представлена на рисунке 2.2.1.

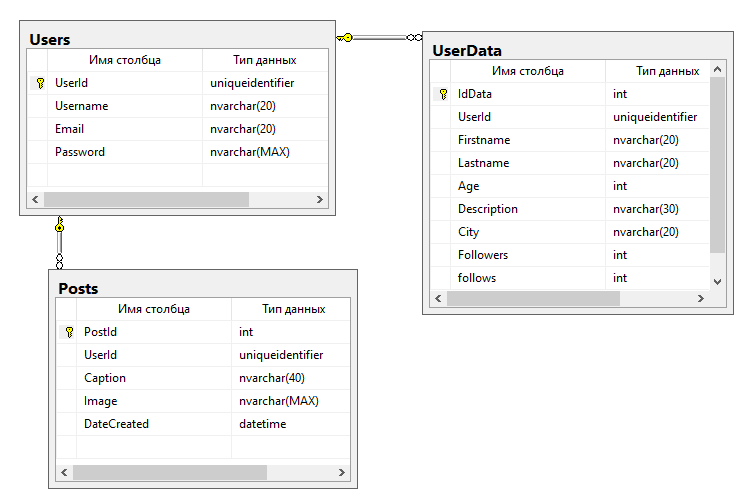


Рисунок 2.2.1 – Схема базы данных

Таким образом, таблица Users содержит ключевое значение UserId, которое используется в остальных таблицах для доступа к данным.

# **2.3 Проектирование сервера**

Для синхронизации двух и более клиентов необходимо серверное приложение. Серверная часть приложения должна предоставлять WEB API для доступа к каждой сущности в базе данных. Передаваемые или получаемые данные от сервера должны передаваться в формате JSON.

Серверное приложение должно быть построено на основе архитектурного стиля REST и будет использоваться для взаимодействия компонентов распределённого приложения в сети.

Сервер содержит в себе реализованные сервисы для взаимодействия с каждой сущностью базы данных.

При обращении к серверу по заданному маршруту и определенному HTTP методу сервер будет определять с какой сущностью базы данных необходимо взаимодействовать и какие действия над данными необходимо будет произвести.

При помощи HTTP метода GET сервер будет возвращать данные клиенту, ссылаясь на аргументы в запросе, как на условие для выборки данных. Метод POST в свою очередь позволяет передать данные на сервер для последующей обработки. Метод PUT позволяет обновить соответствующие данные, переданные с клиента. Метод DELETE удаляет с базы данных заданную запись.

# **2.4 Выводы по главе «Проектирование программного продукта»**

В данной главе было рассмотрено проектирование каждого отдельного компонента приложения.

По части мобильного приложения был определен основной функционал приложения, также были разработаны диаграмма вариантов использования приложения и описаны варианты использования программного средства.

Была также спроектирована база данных, где были определены необходимые таблицы, охарактеризованы поля каждой из них, определены типы данных и ограничения целостности. Также были установлены связи между таблицами, заключающиеся в создании первичных и внешних ключей.

При проектировании серверного приложения был определен архитектурный стиль взаимодействия между клиентом и сервером, формат передачи данных, обозначили HTTP методы которые будут использованы в работе с серверной частью.

# **3 Программная реализация приложений**

# **3.1 Технические средства для разработки**

Для разработки приложений курсового проекта были использованы следующие технические средства:

SQLite – компактная встраиваемая реляционная база данных.

MSSQL Server 2012 – система управления реляционными БД.

MSSQL Server Managment Studio 2012 – СУБД MSSQL Server.

SQL – декларативный язык программирования, применяемый для создания, модификации, и управления данными в РБД.

Android Studio – интегрированная среда разработки для работы с платформой Android.

JetBrains Rider – кроссплатформенная интегрированная среда разработки программного обеспечения для большей части типов проектов .NET, основанная на IntelliJ и плагине ReSharper.

Java – строго типизированный объектно-ориентированный язык программирования.

C# — объектно-ориентированный язык программирования.

.NET Core – это модульная платформа для разработки программного обеспечения с открытым исходным кодом.

Entity Framework – специальная объектно-ориентированная технология на базе фреймворка .NET для работы с данными.

Retrofit – REST клиент для android и Java от компании Square. Он может относительно легко получать и разбирать JSON (или другие структуированные данные) через вебсервисы, использующие REST.

Postman – набор инструментов тестирования API.

Ngrok – это платформа, которая с помощью установленной утилиты, позволяет организовать удалённый доступ на веб-сервер или какой-либо другой сервис, запущенный ПК.

# **3.2 Разработка баз данных**

Для хранения данных на телефоне будет использоваться встраиваемая СУБД SQLite. Для сервера будет использована СУБД MSSQL Server Management Studio.

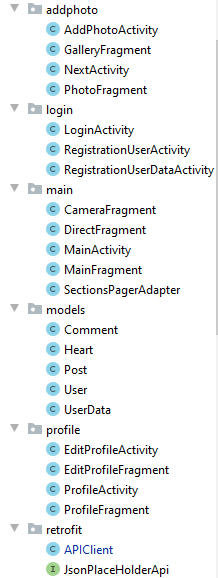
Код создания структуры мобильной базы данных представлен в Приложении Б, серверной в Приложении В.

Мобильная база данных содержит в себе информацию об авторизованном пользователе и используется для проверки, авторизован ли пользователь в приложении.

# **3.3 Разработка мобильного приложения**

Мобильное приложение было реализовано в среде разработки – Android Studio 3.6.3. Для написания приложения был использован язык программирования Java.

Графическая структура проекта представлена на рисунке 3.3.1.



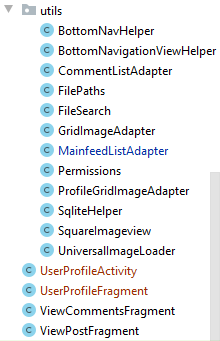


Рисунок 3.3.1 – Графическая структура проекта

Пакет addphoto хранит в себе классы активностей и фрагментов, которые позволяют выбирать и публиковать фото на сервер (см. таблицу 3.3.1).

**Таблица 3.3.1 – Содержание пакета addphoto**

|  |  |  |
| --- | --- | --- |
| Класс AddPhotoActivity – проверка разрешений, инициализация фрагментов | | |
| setupViewPager() | Метод инициализирует фрагменты, использующиеся в данной активности | |
| checkPermissions() | Проверка разрешений на доступ к внешней памяти | |
| Класс GalleryFragment – отображение директориев с фото и их содержимого | | |
| init() | Доступ к директориям и установление их содержимого в GridView | |
| setUpGridview() | Установка содержимого в GridView | |
| setImage() | Установка выбранного фото для публикации | |
| Класс NextActivity – добавление подписи и публикация на сервер, отправка уведомления в случае успеха | | |
| addPhoto() | | POST-запрос на сервер, добавляющий публикацию в таблицу POSTS |
| notification() | | Отправка уведомления в случае успешной публикации |

Пакет login хранит в себе классы активностей, которые отвечают за регистрацию и авторизацию пользователя (см. таблицу 3.3.2).

**Таблица 3.3.2 – Содержание пакета login**

|  |  |
| --- | --- |
| Класс LoginActivity – авторизация пользователей | |
| checkCredentials() | Проверка введенных username&password для авторизации |
| Класс RegistrationUserActivity – регистрация пользователя | |
| addUser() | POST-запрос, отправляющий данные пользователя для регистрации на сервер |
| isEmailValid() | Валидация ввода email |
| Класс RegistrationUserDataActivity – добавление данных пользователя | |
| addUserData() | POST-запрос, отправляющий данные пользователя на сервер |

Пакет main хранит в себе классы для отображения новостной ленты (см. таблицу 3.3.3).

**Таблица 3.3.3 – Содержание пакета main**

|  |  |
| --- | --- |
| Класс MainActivity – проверяет, авторизован ли уже пользователь | |
| isUserLoggedIn() | Проверка наличия данных пользователя в локальной БД |
| Класс MainFragment – отображение публикаций пользователей | |
| getPosts() | GET-запрос для получения публикация пользователей |
| displayPhotos() | Применение адаптера для отображения публикаций с использованием соответствующей разметки |

Пакет models хранит в себе модели классов сущностей (см. таблицу 3.3.4).

**Таблица 3.3.4 – Содержание пакета models**

|  |
| --- |
| Класс Post – класс-модель публикации |
| Класс User – класс-модель данных авторизации пользователя |
| Класс UserData – класс-модель данных пользователя |

Пакет profile хранит в себе классы активностей и фрагментов, позволяющих взаимодействовать с профилем пользователя (см. таблицу 3.3.5).

**Таблица 3.3.5 – Содержание пакета profile**

|  |  |
| --- | --- |
| Класс EditProfileActivity – обновление данных пользователя | |
| setDataToEt() | Установка существующих данных пользователя в EditText. |
| updateUserData() | PUT-метод на сервер для обновления данных пользователя. |
| Класс ProfileActivity – инициализация фрагмента для отображения разметки профиля и слушатель событий | |
| onGridImageSelected() | Слушатель, обрабатывающий нажатие на фото для отображения конкретной публикации |
| Init() | Инициализация фрагмента |
| Класс ProfileFragment - отображение фото и данных пользователя | |
| downloadImage() | GET-запрос на сервер для получения публикаций пользователя |
| getUserData() | GET-запрос на сервер для получения информации о пользователе |
| setupGridView() | Установка фото в GridView |
| stringToImage() | Конвертация строкового представления в Bitmap |

Пакет retrofit содержит в себе конфигурацию для использования Retrofit 2 в запросах на сервер(см. таблицу 3.3.6).

**Таблица 3.3.6 – Содержание пакета retrofit**

|  |
| --- |
| Класс APIClient – конфигурация Http-клиента |
| Интерфейс JsonPlaceHolderApi – определение методов для запросов к серверу |

Пакет utils содержит в себе вспомогательные классы, использующиеся для оптимизации работы и отображения контента в активностях и фрагментах. (см. таблицу 3.3.7).

**Таблица 3.3.7 – Содержание пакета utils**

|  |  |
| --- | --- |
| Класс BottomNavHelper – установка нижнего меню | |
| setUpBottomNavigationView() | Позволяет установить нижнее меню на экран |
| Класс BottomNavigationViewHelper – инициализация нижнего меню | |
| enableNavigation() | Обработка нажатий на пункты меню |
| Класс FilePaths – хранилище констант путей директориев с фото | |
| Класс FileSearch – получение путей директориев и путей файлов фото | |
| Класс GridImageAdapter – адаптер для отображения фото в GridView при выборе фото для публикации | |
| Класс MainFeedListAdapter – адаптер для отображения публикаций в новостной ленте | |
| Класc Permissions – класс-хранилище констант для разрешений приложения | |
| Класс ProfileGridImageAdapter – адаптер для отображения фото в GridView для отображения | |
| Класс SqliteHelper – класс-помощник для работы с локальной БД | |
| onCreate() | Метод, срабатывающий при создании БД |
| onUpdate() | Метод, срабатывающий при обновлении БД |
| insertData() | Вставка данных в БД |
| deleteData() | Удаление данных из БД |
| getData() | Получение данных из БД |
| Класс UniversalImageLoader – класс, отвечающий за установку изображений в ImageView | |

Также, имеются дополнительные классы, не лежащие в пакетах (см. таблицу 3.3.8).

**Таблица 3.3.8 – Дополнительные классы**

|  |
| --- |
| Класс UserProfileActivity – используется для отображения профиля других пользователей приложения, аналогичен ProfileActivity |
| Класс UserProfileFragment – используется для отображения профиля других пользователей приложения, аналогичен ProfileFragment |
| Класс ViewPostFragment – используется для отображения разметки конкретной публикации |

Ознакомиться с диаграммой классов разработанного приложения можно в приложении А.

# **3.4 Разработка сервера**

Сервер было реализовано в среде разработки – JetBrains Rider. Для написания приложения была использована технология ASP.NET WEB API на языке C#.

Графическая структура проекта представлена на рисунке 3.4.1.

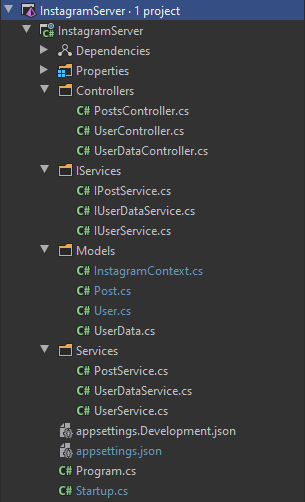


Рисунок 3.4.1 – Графическая структура проекта

Папка Controllers хранит в себе классы контроллеров, обрабатывающих запросы, поступающие к серверу(см. таблицу 3.4.1).

**Таблица 3.4.1 – Содержание папки Controllers**

|  |  |
| --- | --- |
| Класс PostsController – контроллер таблицы публикаций(постов) | |
| Get() | Получение всех постов |
| GetById() | Получение поста по ID пользователя |
| AddPost() | Добавление нового поста |
| DeletePost() | Удаление поста |
| Класс UserController – контроллер таблицы пользователей | |
| GetById() | Получение пользователя по ID |
| Post() | Добавление нового пользователя |
| Delete() | Удаление пользователя |
| Credentials() | Проверка данных авторизации пользователя |
| Класс UserDataController – контроллер таблицы продукт в рецепте | |
| GetById() | Получение данных пользователя по ID |
| Post() | Добавление данных пользователя |
| Put() | Обновление данных пользователя |

Папка IServices содержит интерфейсы, используемые как сервисы.

Папка Services содержит классы, реализующие соответствующие сервисы. Методы данных классов по реализуемой цели соответствуют методам в контроллерах, использующих данные сервисы (см. таблицу 3.4.1).

Папка Models содержит в себе классы-модели, соответствующие сущностям базы данных и класс контекста, обеспечивающий доступ к базе данных MSSQL Server.

Классы, не принадлежащие никакой папке, представлены в таблице 3.4.2. и представляют собой классы конфигурации, которые содержат в себе строки подключения сервисов и строку подключения к БД.

**Таблица 3.4.2 – Основные классы**

|  |
| --- |
| Класс Program – определяет начальную точку входа в приложение |
| Класс Startup – определяет конфигурацию приложения |

# **3.5 Вывод по главе «Программная реализация приложений»**

В данной главе было рассмотрена разработка каждого отдельного компонента приложения.

На первоначальном этапе были рассмотрены технологии, которые применены для разработки курсового проекта.

Вторым этапом была разработка структуры базы данных на стороне сервера и клиента.

Третьим этапом стала разработка мобильного приложения, где была представлена графическая структура приложения, рассмотрен каждый класс и их методы, в частности.

Четвертый этап – это разработка серверного приложения, где была представлена графическая структура приложения, рассмотрен каждый класс и их методы.

# **4. Руководство пользователя**

Для того, чтобы установить приложение на телефон, Вам необходимо иметь версию Android 7.0 и выше(версию API 22 и более). Версию Вашего телефона можно узнать в настройках.

Если Ваша версия удовлетворяет требованиям, то можно устанавливать приложение на телефон.

После установки приложения на экране появится иконка и подпись “Instagram” (рис.4.1)

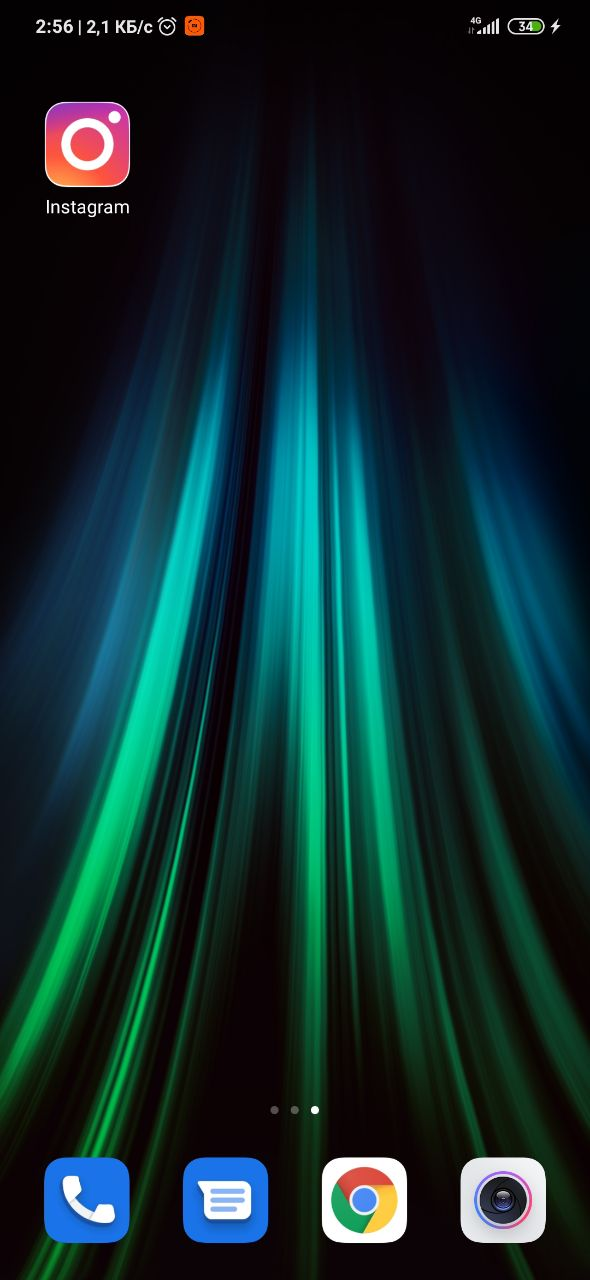


Рисунок 4.1 – Иконка приложения

Изначально открывается экран, на котором находится форма входа(рис.4.2).

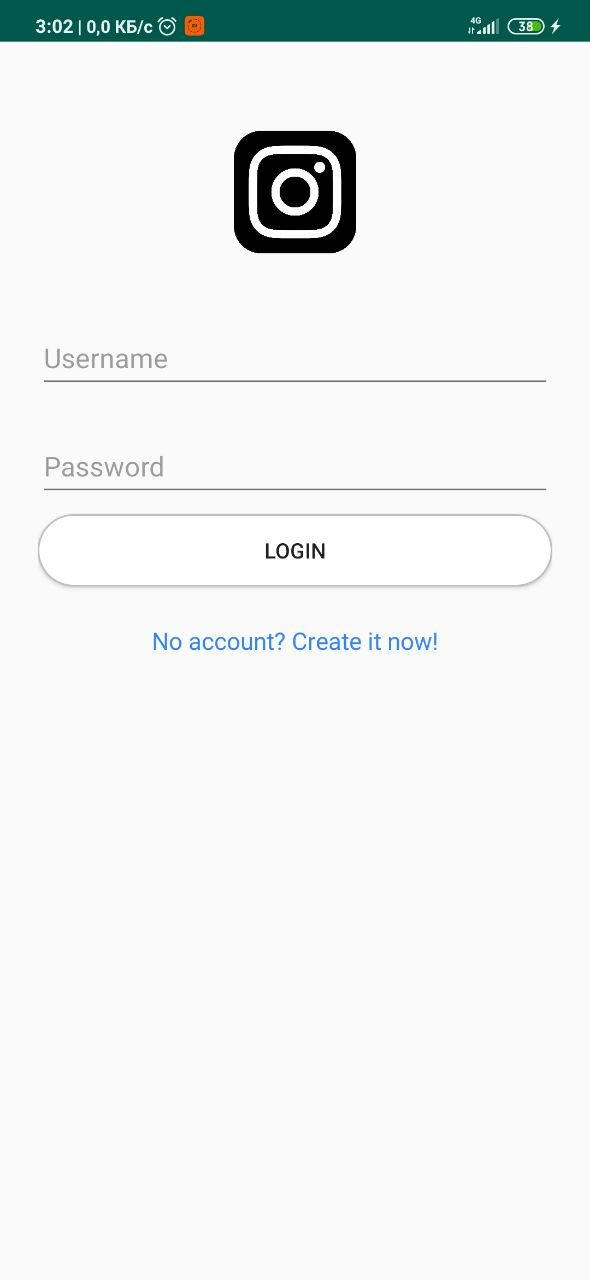


Рисунок 4.2 – Форма входа

Перед первым использованием приложения Вам необходимо зарегистрироваться. Для этого нужно нажать на подпись “No account? Create it now!”, в открывшуюся форму ввести свой Email, password и придумать имя пользователя (username). После ввода данных Вам необходимо нажать клавишу ADD INFORMATION и вы автоматически перейдете к заполнению данных пользователя. Вам следует ввести Имя, Фамилию, возраст, город, в котором Вы находитесь и написать краткую информацию о себе. Заполнение данной информации не обязательно. Нажав на клавишу ADD DATE Вы переходите в окно, где находится форма входа.

Вы успешно прошли регистрацию и можете зайти в свой аккаунт. Для этого Вы заполняете username (имя пользователя) и вводите пароль(password) на форме входа, после нажатия клавиши LOGIN Вы переходите в приложение.

Сразу открывается главный экран, в котором отображается загрузка публикаций пользователей. После загрузки Вы можете увидеть сообщение об успешной загрузке и публикации всех пользователей(рис.4.3)

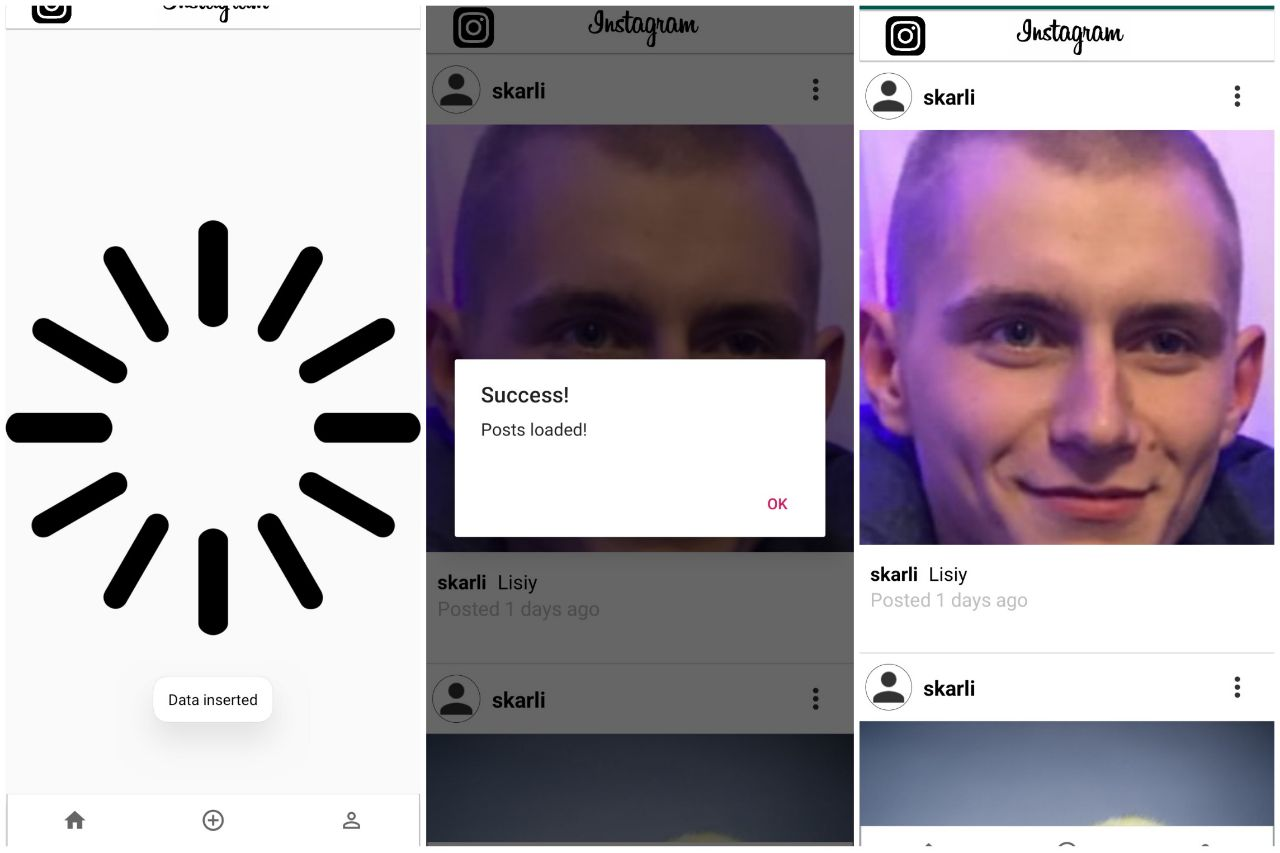


Рисунок 4.3 – Главный экран

При нажатии на фото профиля пользователя в ленте, Вы можете перейти на страницу с его профилем, где можно увидеть его данные и фото, которые опубликованы в приложении этим пользователем(рис.4.4):

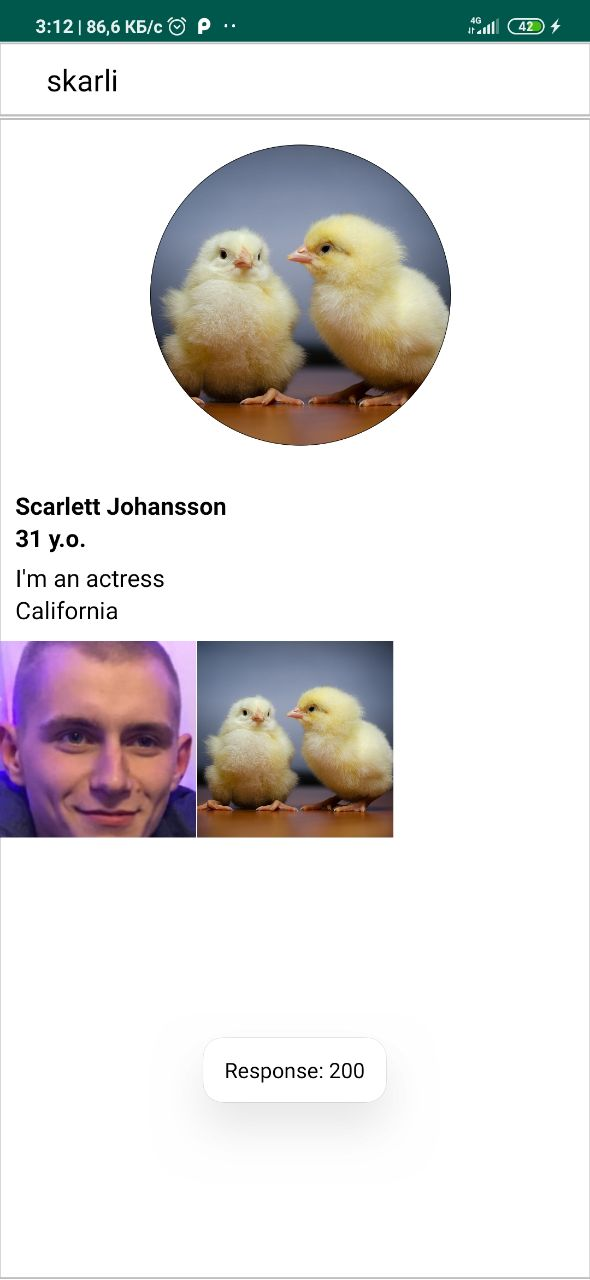


Рисунок 4.4 – Профиль пользователя

В приложении можно разместить, как сохраненные фотографии из галереи в телефоне, так и сразу сделать фото, выложив его в инстаграм. Для загрузки фото из галереи Вам необходимо зайти в свой аккаунт, в нижней части экрана, посередине, расположен знак «+», нажав на него Вы переходите в новую вкладку.

Для доступа к фото вам необходимо одобрить разрешения приложения, необходимые для доступа к вашим файлам(рис.4.5):

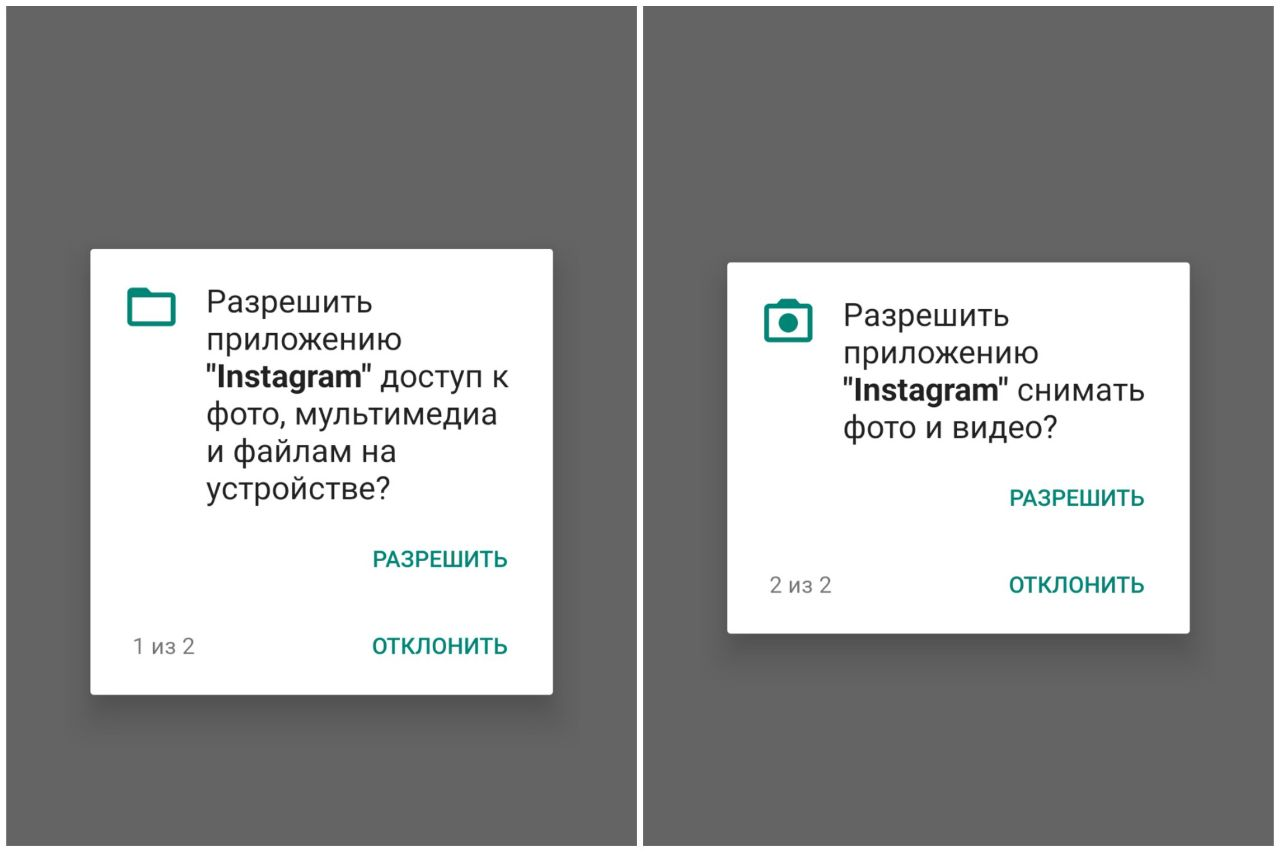


Рисунок 4.5 – Разрешения приложения

Далее Вы можете выбрать фото из гелереи или сделать фото прямо в программе, с камеры устройства.

Для публикации фото из галереи, в верхней части экрана нужно выбрать папку, из которой хотите взять фото и затем выбрать само фото для публикации. Затем нажать “Next” в правом верхнем углу и перейти на страницу, где можно придумать подпись к фото и, наконец, опубликовать фото на сервер(рис.4.6):

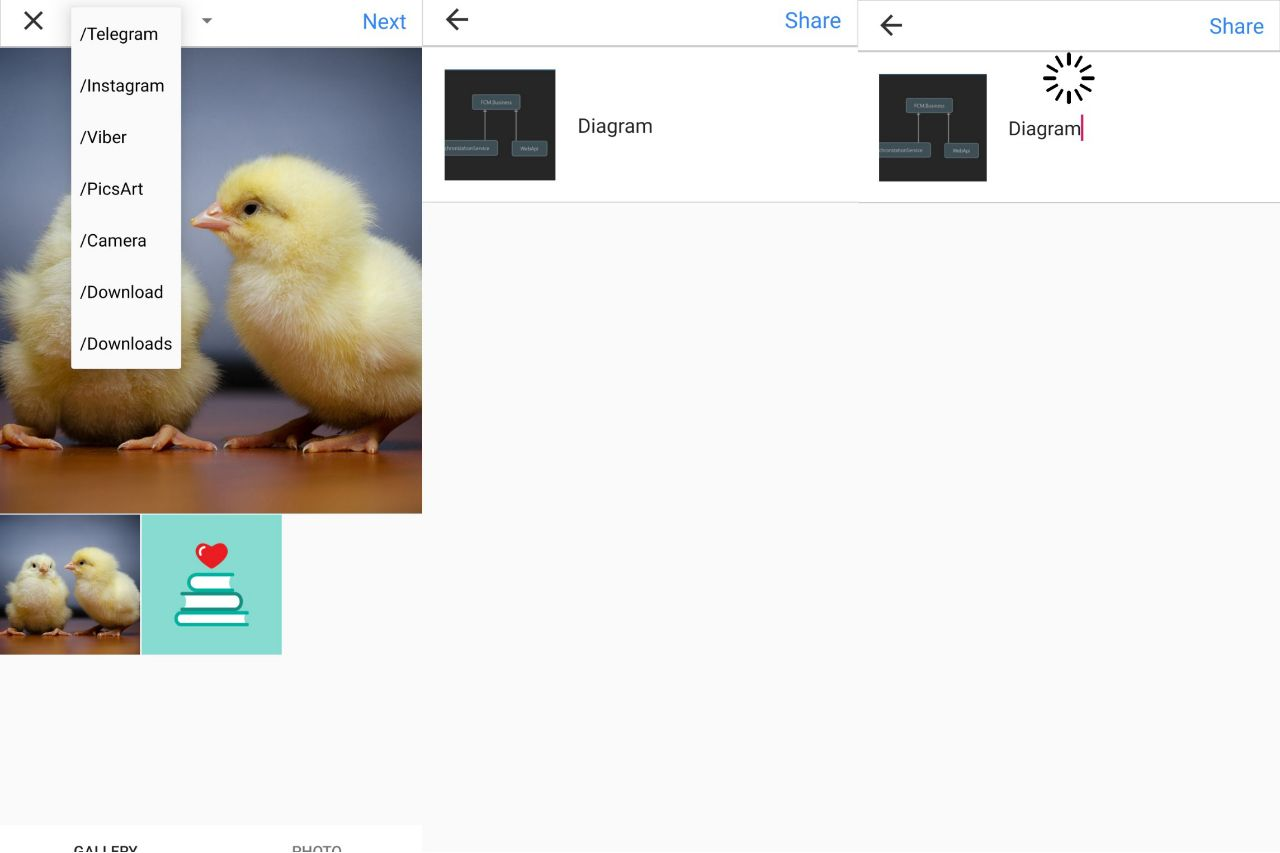


Рисунок 4.6 – Публикация фото из галереи

В случае успешной публикации, на устройство придет соответсвующее уведомление(рис.4.7):

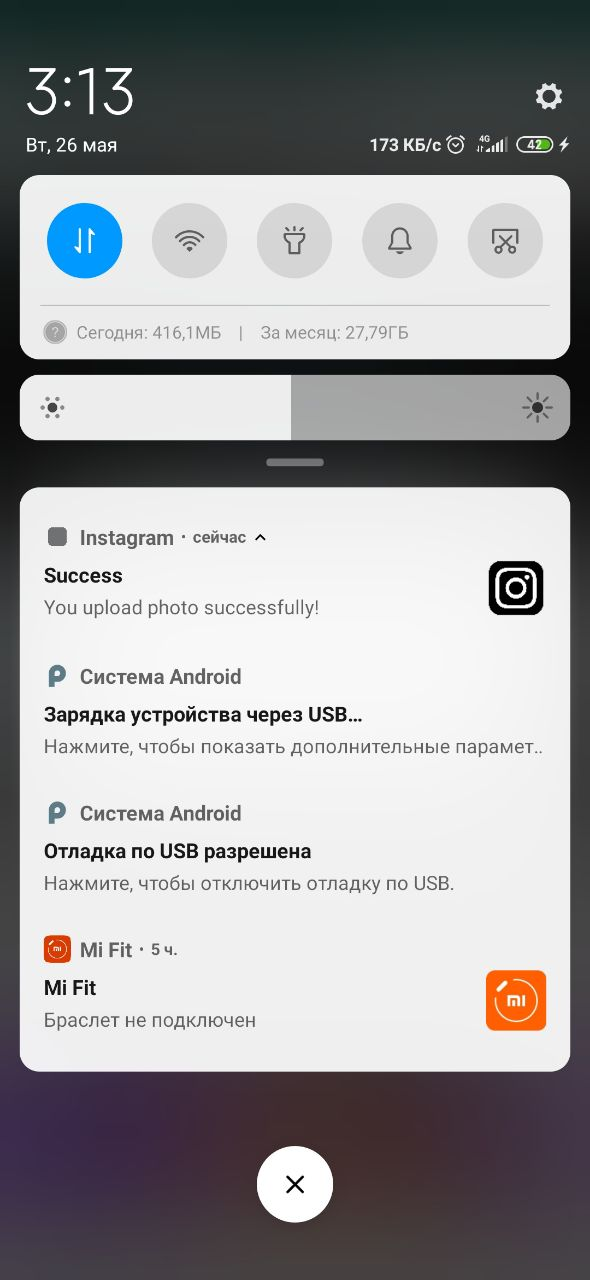


Рисунок 4.7 – Уведомление об успешной публикации

И фото появится в ленте у остальных пользователей(рис.4.8):

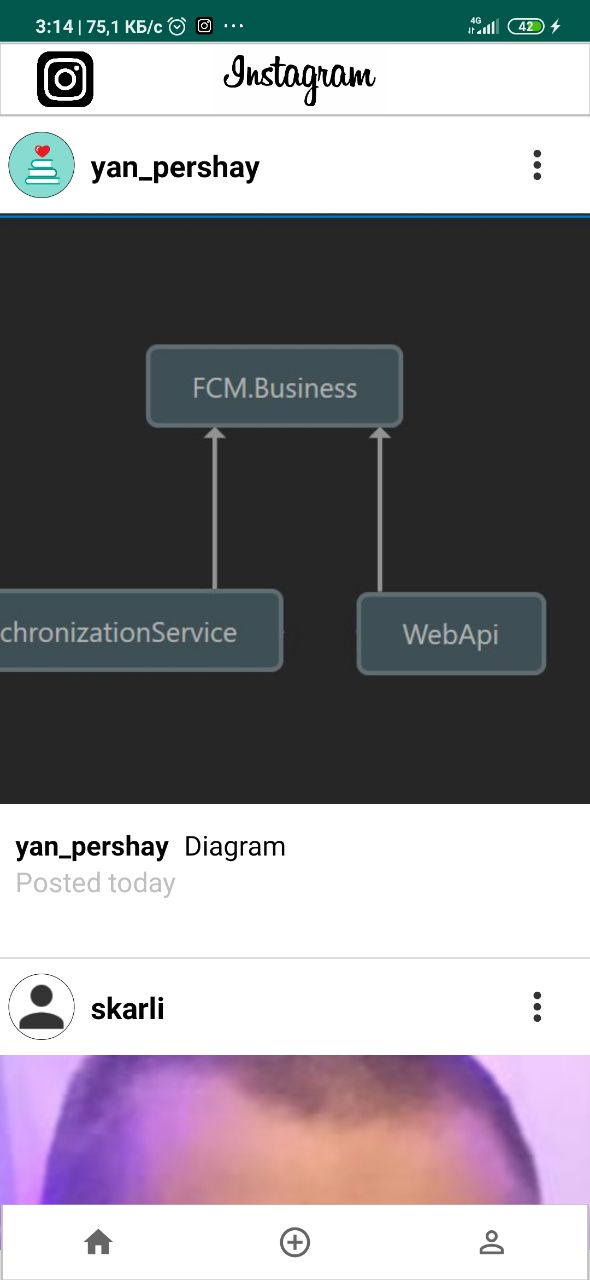


Рисунок 4.8 – Новое фото в ленте

Также есть возможность сделать фото прямо с камеры устройства. Для этого в нижнем меню выбираем “PHOTO”, после нажатия на кнопку “OPEN CAMERA” открывается камера. Делаете снимок и публикация происходит по тому же алгоритму, что и публикация фото с галереи(рис.4.9):

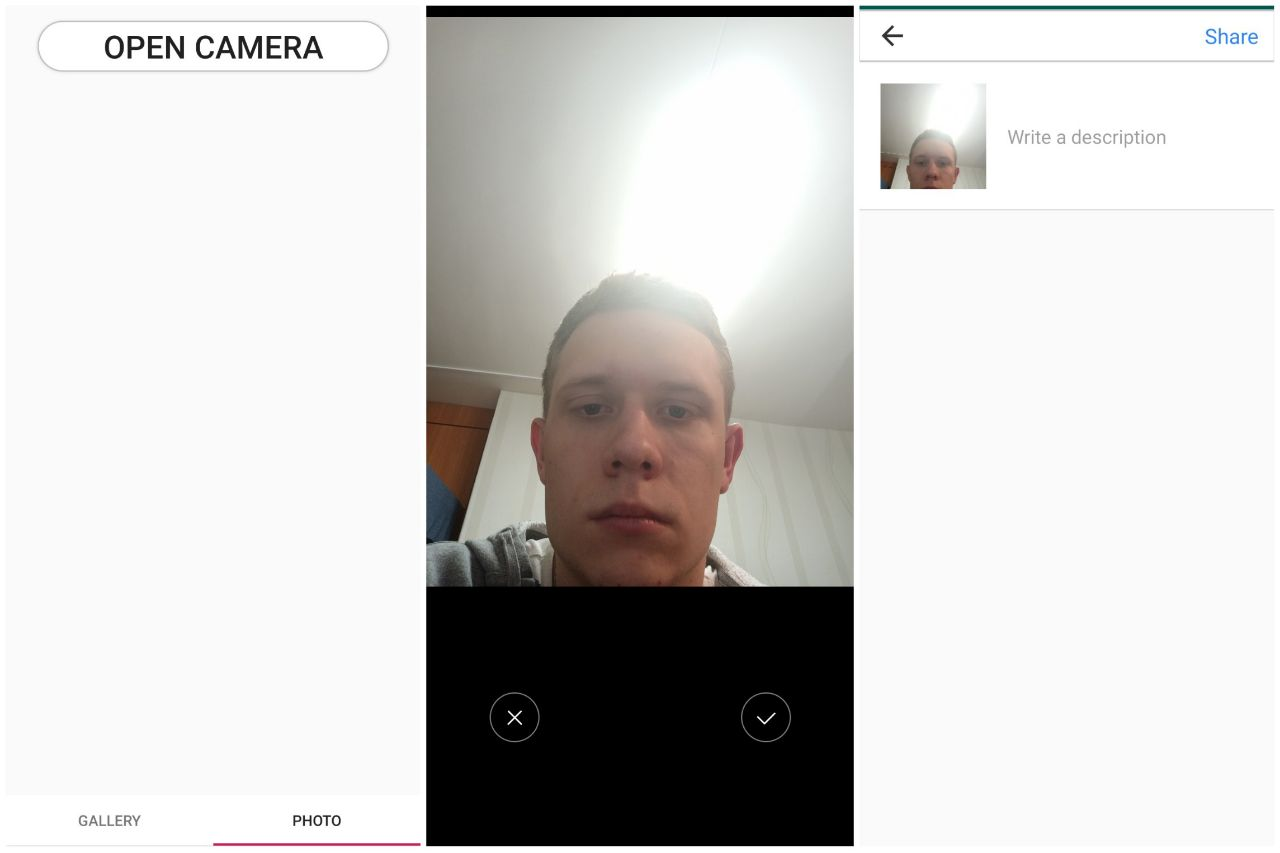


Рисунок 4.9 – Публикация фото с камеры

Для просмотра личного профиля нужно выбрать правую кнопку в нижнем меню. В профиле есть возможность посмотреть личную информацию, отредактировать её, посмотреть свои публикации, а также удалить их. Ниже будет продемонстрировано взаимодействие с личным профилем.

Личный профиль(рис.4.10):

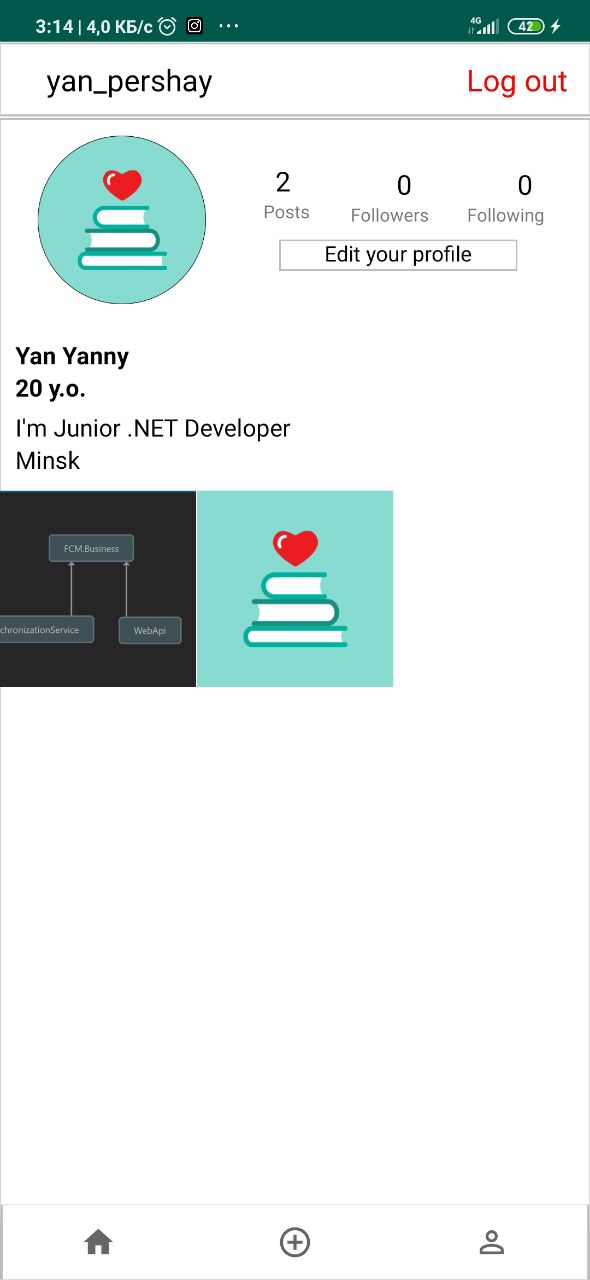


Рисунок 4.10 – Личный профиль

Редактирование собственной информации(рис.4.11). Для редактирования нужно просто поменять соответствующие текстовые поля на нужные Вам значения:

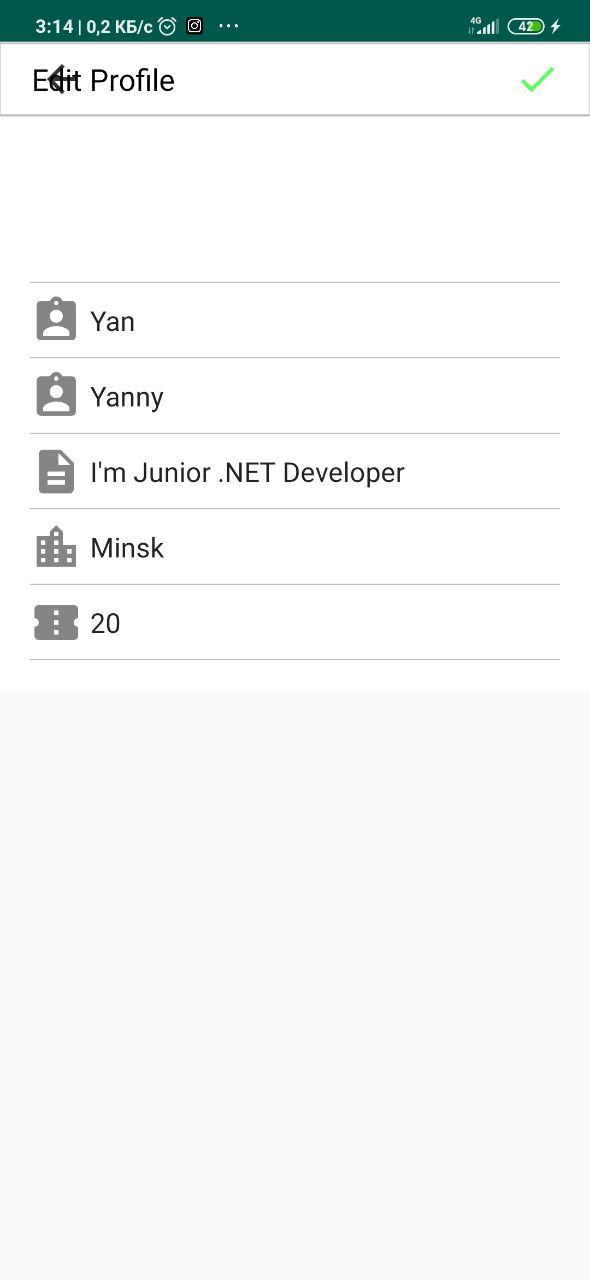


Рисунок 4.11 – Редактирование профиля

Просмотр публикации(рис.4.12). Для просмотра публикации нужно нажать на фото.



Рисунок 4.12 – Просмотр публикации

Удаление публикации(рис.4.13). Для удаления публикации необходимо нажать на значок «красная корзина» и подтвердить удаление.

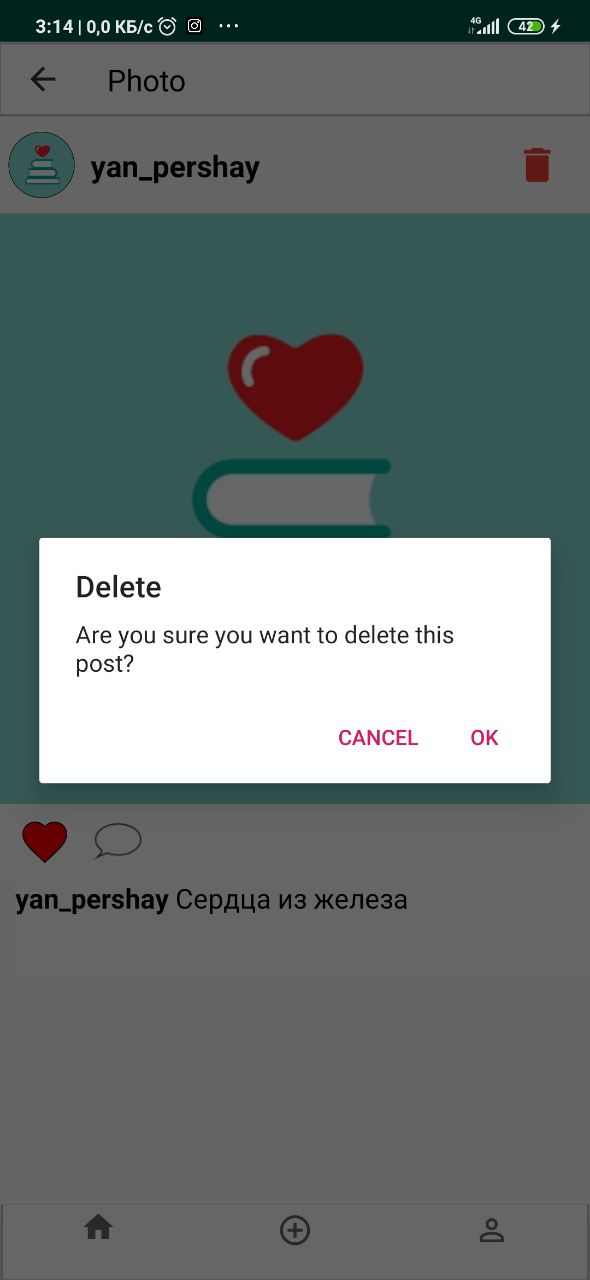


Рисунок 4.13 – Удаление публикации

Также имеется возможность обновления новостной ленты. Для этого нужно потянуть начало ленты вниз, появится значок обратной стрелочки и начнется загрузка ленты(рис.4.14).

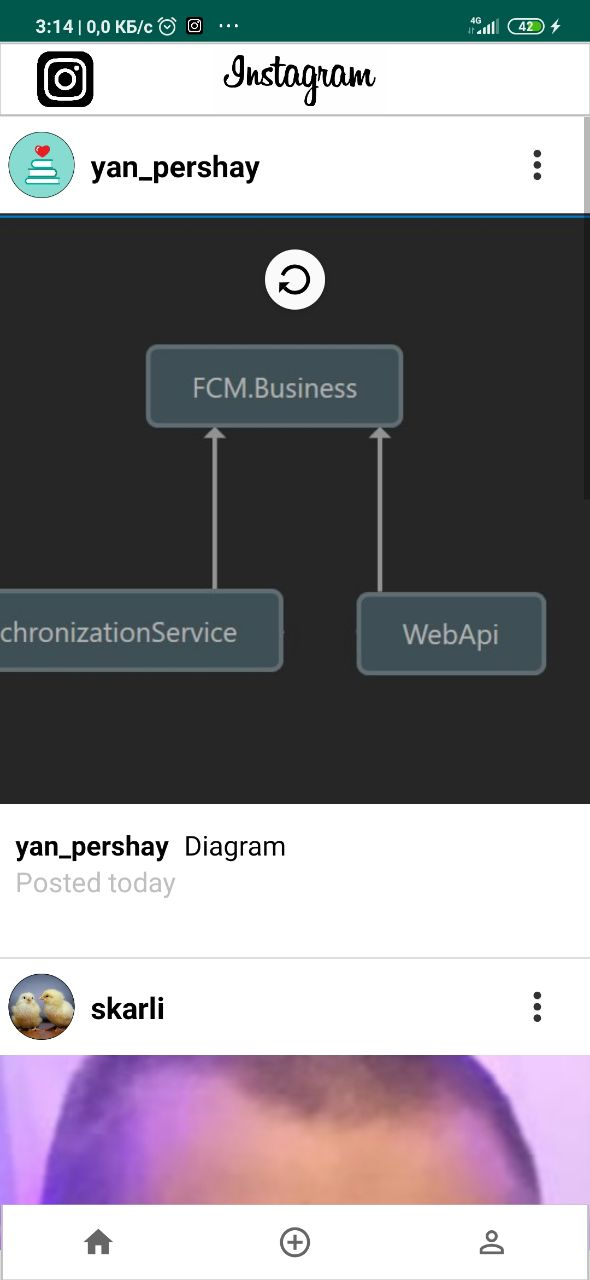


Рисунок 4.14 – Обновление новостной ленты

# **4.1 Выводы по главе «Руководство пользователя»**

В данной главе было описано руководство пользователя.

Данное руководство позволяет пользователю узнать требования для установки приложения на телефон, а также как использовать все функции приложения. Были предоставлены скриншоты самого приложения для большей наглядности использования.

# **ЗАКЛЮЧЕНИЕ**

Целью моего курсового проекта было предоставить пользователям возможность обмена фотографиями.

Были решены следующие поставленные задачи:

* обзор аналогов;

В результате было рассмотрено 2 приложения и были выявлены их положительные и отрицательные стороны. Эта информация была использована для улучшения функционала приложения и пользовательского интерфейса. Таким образом, я не вводил систему лайков, чтобы пользователи больше сосредоточились не на статистике, а не обмене фото.

* проектирование приложения;

Был спроектирован каждый отдельный компонент приложения. В мобильном приложении был определен основной функционал, также была разработана диаграмма вариантов использования. При проектировании базы данных, были определены необходимые таблицы и разработана схема базы данных. В серверном приложении был определен архитектурный стиль взаимодействия между клиентом и сервером, формат передаваемых данных и HTTP методы для работы с серверной частью.

* реализация приложения;

Данная задача была реализована с помощью следующих технических средств: СУБД SQLite и MSSQL Server, сред разработки Android Studio и JetBrains Rider, языков программирования Java и C#, модульной платформы .NET Core, библиотек Entity Framework, Retrofit 2, утилиты Ngrok.

Были реализованы базы данных на стороне сервера и клиента.

При решении данной задачи была достигнута поставленная цель, и был создан проект «Социальная сеть для обмена фото “Instaclone”», который включает в себя android-клиент и сервер.

* разработка руководства пользователя.

Разработано руководство, позволяющее пользователю узнать требования для установки приложения на телефон, а также как использовать все функции мобильного приложения.

Мобильное приложение имеет простой, интуитивно понятный пользователю интерфейс и обладает следующими особенностями:

* Публикация фото из галереи;
* Публикация фото с камеры устройства;
* Удаление публикаций;
* Просмотр и редактирование личного профиля;
* Просмотр профилей других пользователей;
* Отправка уведомлений;
* Шифрование паролей в БД.

В то же время, я считаю, что для улучшения работы приложения и повышения его эффективности и востребованности нужно реализовать следующие задачи:

* Оптимизировать постепенную подгрузку данных с сервера;
* Оптимизировать подгрузку фото путём кеширования;
* Добавить возможность вести диалоги между пользователями;
* Добавить систему подписок между пользователями;
* Поменять способ хранения изображений на облачный.

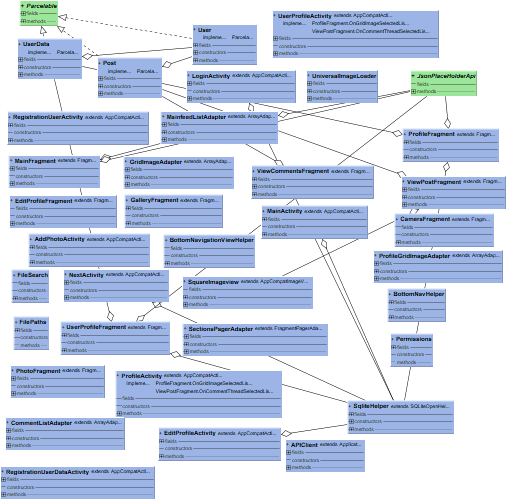
# **Список использованных источников**

1 Build a REST API for your Mobile App with ASP.NET Core [Электронный ресурс]. / stormpath.com – Режим доступа: https://stormpath.com/blog/rest-api-mobile-dotnet-core

1. Retrofit 2. Upgrade Guide [Электронный ресурс] / futurestud.io. – Режим доступа: https://futurestud.io/tutorials/retrofit-2-upgrade-guide-from-1-9
2. Архитектура REST [Электронный ресурс] – habr.com – Режим доступа: https://habr.com/ru/post/38730/
3. Solutions on Stackoverflow [Электронный ресурс] – Режим доступа: https://stackoverflow.com.
4. «Сайт Александра Климова» по Android [Электронный ресурс] – Режим доступа: http://developer.alexanderklimov.ru

**ПРИЛОЖЕНИЕ А**

Диаграмма классов Android-клиента



**ПРИЛОЖЕНИЕ Б**

Код создания структуры БД сервера

CREATE DATABASE INST

GO

USE INST

GO

CREATE TABLE Users(

UserId UNIQUEIDENTIFIER PRIMARY KEY NOT NULL,

Username NVARCHAR(20) NOT NULL,

Email NVARCHAR(20) NOT NULL,

Password NVARCHAR(max) NOT NULL,

Constraint username\_unique Unique(username)

);

CREATE TABLE Posts(

PostId INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

UserId UNIQUEIDENTIFIER NOT NULL,

Caption NVARCHAR(200),

Image nvarchar(max) NOT NULL,

DateCreated DATETIME NOT NULL,

FOREIGN KEY (UserId) REFERENCES Users(UserId)

);

CREATE TABLE UserData(

IdData int identity(1,1) primary key,

UserId UNIQUEIDENTIFIER,

Firstname NVARCHAR(20),

Lastname NVARCHAR(20),

Age INT,

Description NVARCHAR(30),

City NVARCHAR(20),

Followers int not null,

follows int not null,

FOREIGN KEY (UserId) REFERENCES Users(UserId)

);

**ПРИЛОЖЕНИЕ В**

Код хеширования паролей в БД на сервере

#region HashPassword  
  
byte[] salt;  
new RNGCryptoServiceProvider().GetBytes(salt = new byte[16]);  
  
var pbkdf2 = new Rfc2898DeriveBytes(user.Password, salt, 10000);  
byte[] hash = pbkdf2.GetBytes(20);  
  
byte[] hashBytes = new byte[36];  
Array.Copy(salt, 0, hashBytes, 0, 16);  
Array.Copy(hash, 0, hashBytes, 16, 20);  
  
string savedPasswordHash = Convert.ToBase64String(hashBytes);  
  
user.Password = savedPasswordHash;  
  
#endregion